AFRL-RH-FS-TR-2015-0013

Standard Procedure for Calibrating an Areal Calorimetry Based Dosimeter

Charles W. Beason
Devon Ryan

General Dynamics Information Technology

Leland Johnson

Bioeffects Division
Radio Frequency Bioeffects Branch

May 2015

DESTRUCTION NOTICE – Destroy by any method that will prevent disclosure of contents or reconstruction of this document.

Distribution A: Approved for public release; distribution unlimited (P.A. Case No. TSRL-PA-2015-0136, 24 Sep 15).

Air Force Research Laboratory
711th Human Performance Wing
Human Effectiveness Directorate
Bioeffects Division
Radio Frequency Bioeffects Branch
4141 Petroleum Road
JBSA Fort Sam Houston, Texas
78234-2644

NOTICE AND SIGNATURE PAGE

Using Government drawings, specifications, or other data included in this document for any purpose other than Government procurement does not in any way obligate the U.S. Government. The fact that the Government formulated or supplied the drawings, specifications, or other data does not license the holder or any other person or corporation; or convey any rights or permission to manufacture, use, or sell any patented invention that may relate to them.

This report was cleared for public release by the 88th ABW Public Affairs Office and is available to the general public, including foreign nationals. Copies may be obtained from the Defense Technical Information Center (DTIC) (http://www.dtic.mil).

"Standard Procedure for Calibrating an Areal Calorimetry Based Dosimeter"

(AFRL-RH-FS-TR- 2015 - 0013) has been reviewed and is approved for publication in accordance with assigned distribution statement.

JOHNSON.LELAND.
R.1161902612

Digitally signed by
JOHNSON.LELAND.R.1161902612
DN: c=US, o=U.S. Government, ou=DoD, ou=PKI,
ou=USAF, cn=JOHNSON.LELAND.R.1161902612
Date: 2015.08.03 08:39:25 -05'00'

LELAND JOHNSON, DR-III, DAF
Contract Monitor
Radio Frequency Bioeffects Branch

MILLER.STEPHANI
E.A.1230536283

Digitally signed by
MILLER.STEPHANIE.A.1230536283
DN: c=US, o=U.S. Government, ou=DoD, ou=PKI,
ou=USAF, cn=MILLER.STEPHANIE.A.1230536283
Date: 2015.09.17 16:05:15 -05'00'

STEPHANIE A. MILLER, DR-IV, DAF
Chief, Bioeffects Division
Human Effectiveness Directorate
711th Human Performance Wing
Air Force Research Laboratory

This report is published in the interest of scientific and technical information exchange, and its publication does not constitute the Government's approval or disapproval of its ideas or findings.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)* 22-05-2015	**2. REPORT TYPE** Interim	**DATES COVERED** *(From – To)* 15 July 2011 – 1 June 2012
4. TITLE AND SUBTITLE Standard Procedure for Calibrating an Areal Calorimetry Based Dosimeter		**5a. CONTRACT NUMBER** FA8650-13-D-6368-0002 **5b. GRANT NUMBER** N/A **5c. PROGRAM ELEMENT NUMBER** 62202F
6. AUTHOR(S) Charles W. Beason, Devon Ryan, and Leland Johnson		**5d. PROJECT NUMBER** N/A **5e. TASK NUMBER** N/A **5f. WORK UNIT NUMBER** HOEB
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) General Dynamics Information Technology and Air Force Research Laboratory 711th Human Performance Wing, Human Effectiveness Directorate, Bioeffects Division, Radio Frequency Bioeffects Branch (711 HPW/RHDR) 4141 Petroleum Road JBSA Fort Sam Houston, Texas 78234-2644		**8. PERFORMING ORGANIZATION REPORT NUMBER** N/A
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) Air Force Materiel Command, Air Force Research Laboratory, 711th Human Performance Wing, Human Effectiveness Directorate, Bioeffects Division, Radio Frequency Bioeffects Branch (711 HPW/RHDR) 4141 Petroleum Road JBSA Fort Sam Houston, Texas 78234-2644		**10. SPONSOR/MONITOR'S ACRONYM(S)** 711 HPW/RHDR **11. SPONSOR/MONITOR'S REPORT NUMBER(S)** AFRL-RH-FS-TR-2015-0013

12. DISTRIBUTION / AVAILABILITY STATEMENT
Distribution A: Approved for public release; distribution unlimited (P.A. Case No. TSRL-PA-2015-0136, 24 Sep 15).

13. SUPPLEMENTARY NOTES
This report supercedes AFRL-RH-FS-TR-2014-0001.

14. ABSTRACT

In this report, a methodology is presented for calibrating an areal calorimetry-based dosimeter. A detailed example of calibrating an areal calorimetry-based dosimeter is provided to assist in calibrating other dosimeters.

15. SUBJECT TERMS
Millimeter waves, dosimeter, calorimeter, CLT, Carbon-loaded Teflon

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** L. Johnson
a. REPORT Unclassified	**b. ABSTRACT** Unclassified	**c. THIS PAGE** Unclassified	SAR	66	**19b. TELEPHONE NUMBER** (include area code) N/A

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39.18

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ACRONYMS

711 HPW/RHDR	Air Force Research Laboratory, 711th Human Performance Wing, Human Effectiveness Directorate, Bioeffects Division, Radio Frequency Bioeffects Branch
CL	Center Left
CLT	Carbon-loaded Teflon
CR	Center Right
GPIB	General Purpose Interface Bus
IR	Infrared
LC	Lower Center
LL	Lower Left
LR	Lower Right
MMW	Millimeter Wave
OEWG	Open Ended Waveguide
PRF	Pulse Repetition Frequency
RF	Radio Frequency
ROI	Regions of Interest
SGH	Standard Gain Pyramidal Horn
UC	Upper Center
UL	Upper Left
UR	Upper Right

EXECUTIVE SUMMARY

In this report, a methodology is presented for calibrating an areal calorimetry-based dosimeter. A detailed example of calibrating an areal calorimetry-based dosimeter is provided to assist in calibrating other dosimeters.

1.0 INTRODUCTION

In this report, a methodology is presented for calibrating an areal calorimetry-based dosimeter. Since 2002, the Air Force Research Laboratory, 711th Human Performance Wing, Human Effectiveness Directorate, Bioeffects Division, Radio Frequency Bioeffects Branch (711 HPW/RHDR) has used carbon-loaded Teflon® (CLT) as the radio frequency (RF) absorber for the dosimeter. The methodology presented will use CLT for the calibration example. However, the calibration methodology presented is identical for any other areal calorimetry-based dosimeter which 711 HPW/RHDR intends on developing.

2.0 BACKGROUND

2.1 Theoretical Basis

In a homogeneous and thermally conductive material, heat is diffused by conduction through the material. In one dimension, this diffusion is governed by the equation:

$$\partial_{z,z}T(z,t) = \frac{1}{\kappa}\,\partial_t T(z,t) \tag{1}$$

where $T(z,t)$ is the media temperature at position z and time t and $\frac{1}{\kappa}$ is the media density ρ times the specific heat C divided by the thermal conductivity K.

Assume a semi-infinite slab of millimeter wave (MMW)-absorbing material existing in the positive z direction, with MMWs impinging on the material. The heat from this radiation will be absorbed according to

$$Q(z) = \frac{P_i\,\tau}{\delta\,\mathrm{K}} e^{\frac{-z}{\delta}} \tag{2}$$

where $Q(z)$ is the heat deposited at position z, P_i is the incident RF power, τ is the proportion of the MMW power that is absorbed, and δ is the energy deposition depth.

The heat equation for the slab is thus:

$$\partial_{z,z}T(z,t) + \frac{P_i\,\tau}{\delta\,\mathrm{K}} e^{\frac{-z}{\delta}} = \frac{1}{\kappa}\,\partial_t T(z,t) \tag{3}$$

The modeled boundary conditions are:

$$T(\infty,t) = 0\,, \qquad t \geq 0\,. \tag{4}$$

At the surface z=0, it is assumed that heat is lost to the environment at a rate that is proportional to the difference between the surface temperature T(0,t) and the environmental temperature, T_E:

$$\partial_z T(0,t) = \alpha(T(0,t) - T_E), \tag{5}$$

where α is the coefficient of surface heat transfer. The time-dependent solution for the heat distribution in the slab is:

$$\begin{aligned} T(z,t) = (T_E - T_0) \\ &+ \frac{1}{\mathrm{K}}\Bigg(P_i\,\delta\,\tau\Bigg(e^{\frac{-z}{\delta}}\left(1 - \frac{1}{2}e^{\frac{-\kappa t}{\delta^2}}\,Erfc\left[\frac{2\,\kappa\,t - z\,\delta}{2\,\delta\,\sqrt{\kappa\,t}}\right]\right) \\ &+ \frac{1}{\alpha\,\delta\,(\alpha\,\delta - 1)}\Bigg((1 - \alpha^2\,\delta^2)\,Erfc\left[\frac{z}{2\,\sqrt{\kappa\,t}}\right] \\ &- e^{\alpha\,(z+\kappa\,t\,\alpha)}Erfc\left[\frac{2\,\kappa\,t\,\alpha + z}{2\,\sqrt{\kappa\,t}}\right] \\ &+ \frac{1}{2}e^{\frac{\kappa\,t+z\,\delta}{\delta^2}}\,\alpha\,\delta(1+\alpha\,\delta)Erfc\left[\frac{2\,\kappa\,t + z\,\delta}{2\,\delta\,\sqrt{\kappa\,t}}\right]\Bigg)\Bigg)\Bigg) \end{aligned} \tag{6}$$

The infrared (IR) energy detected by the camera is assumed to originate from the surface of the CLT, and therefore, only the solution at the surface (z = 0) is of interest:

$$\begin{aligned} T(0,t) = (T_E - T_0) \\ &+ \frac{1}{\mathrm{K}}\Bigg(P_i\,\delta\,\tau\Bigg(\left(1 - \frac{1}{2}e^{\frac{\kappa\,t}{\delta^2}}\,Erfc\left[\sqrt{\frac{\kappa\,t}{\delta^2}}\right]\right) \\ &+ \frac{1 - \alpha^2\,\delta^2 - e^{\kappa\,t\,\alpha^2}Erfc\left[\alpha\sqrt{\kappa\,t}\right]}{\alpha\,\delta\,(\alpha\,\delta - 1)} \\ &+ \frac{e^{\frac{\kappa\,t}{\delta^2}}\,(1+\alpha\,\delta)Erfc\left[\sqrt{\frac{\kappa\,t}{\delta^2}}\right]}{2(\alpha\,\delta - 1)}\Bigg)\Bigg) \end{aligned} \tag{7}$$

Generating a fit to this equation proved to be problematic. This solution complexity was the major reason a "semi-empirical" fit for modeling skin (and CLT) was chosen. The form of the equation used to fit is (Beason, 2005)

$$T_{Surface}(t,p) = T_0 + t\,\beta\,p\,e^{-\gamma\,t} + p\left(T_{Equib} - T_0\right)\left(1 - e^{-\epsilon\,t}\right) \qquad (8)$$

where:

$T_{Surface}(t,p)$	is temperature at the surface and is a function of time and power,
t	is time relative to when MMW exposure commences,
p	is the incident power,
T_{Equib}	is the equilibrium temperature reached at large t, and

β, γ, and ϵ are constants to be fit. This semi-empirical equation is valid for times during a single MMW exposure at a constant power density.

3.0 ACQUISITION OF CALIBRATION DATA

3.1 Experimental Setup

Data are typically acquired in an experimental setup similar to the one illustrated in Figure 1. In the setup depicted, a dielectric lens is utilized. This lens is only necessary if either very small spots or very high powers are required for the calibration. MMWs from the source pass through a dielectric lens that focuses the MMWs into a small Gaussian spot. An IR camera is used to acquire thermal images of the spot on the CLT as it is heated. The peak temperature of the spot for each frame is recorded in a text file by the data acquisition program. A plot of some example data recorded in the file is shown in Figure 2.

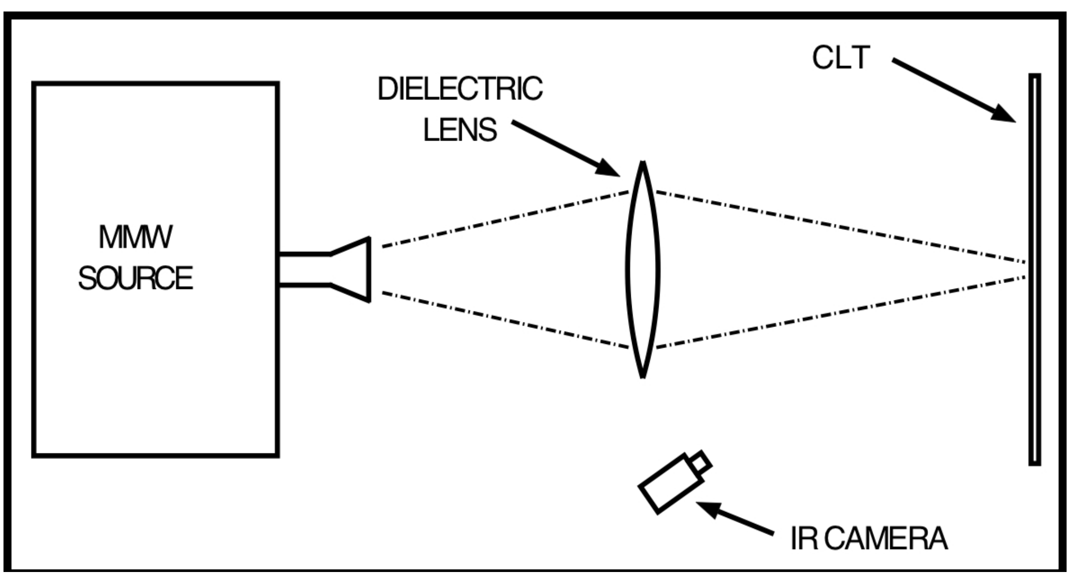


Figure 1. Calibration setup.

In Figure 2, note neither the time nor temperature start at zero. The initial flat portion of the data is the temperature of the CLT before the radio frequency comes on and impinges upon the detector target surface. In this case, the source was on for approximately 2.5 s, shortly after which the data acquisition ends.

For this shot, the power was set to 4 W/cm^2. The peak dose on the CLT was 10 J/cm^2. Typically, the total peak dose should be at least 10 J/cm^2, but never over 15 J/cm^2. This is so the CLT heats sufficiently to achieve a temperature increase well above the thermal noise while protecting the CLT from overheating. This rule should only be ignored when obtaining the equilibrium data.

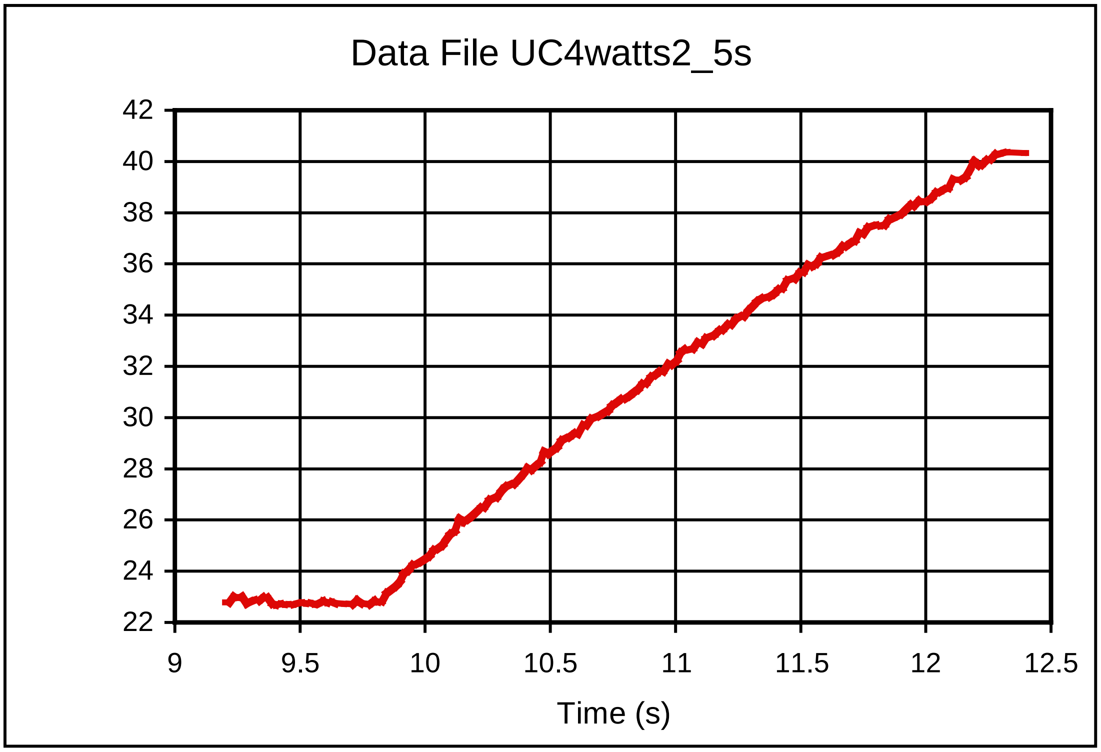


Figure 2. Plot of data from IR camera data acquisition program.

A series of shots and powers is necessary for CLT calibration. At least four powers should be used. The lowest should be at a level that achieves a temperature increase well above the thermal noise within the dose limitation. For the CLT in use, a power of 0.5 W/cm^2 has been found to work quite well. The highest power used should be at least the maximum power for which the CLT is expected to be used. Outdoor measurements rarely exceeded 6 W/cm^2; however, in the laboratory, powers as high as 9 W/cm^2 have been used. The number of shots at each power should be sufficient to determine if the MMW source is behaving consistently. Averages of eight shots are sufficient, but more is always better.

If possible, all shots should be taken on CLT that has cooled to approximately background temperature. Also, the shots for each power should be at several different locations on the CLT. This approach will provide data regarding the homogeneity of CLT.

The equilibrium shots should be done for at least two powers if possible because the equilibrium temperature is linear with power for our parameter space. It is possible that time constraints may preclude finding the equilibrium at more than one power. As before, more data generated is always better.

3.2 Experimental Procedure

Listed below are steps to calibrate CLT:

1. Power up 94 GHz RF source and prepare for operation in accordance with Standard Operating Procedure for Northstar 94 GHz Radio Frequency Source (Mylacraine, 2013)

2. Configure IR cameras

   Infrared cameras operate at different wavelength bands. The emissivity of the CLT also varies across these bands. If a different camera is being used in the field than the camera used for the calibration, emissivity becomes an important consideration and the emissivity of the CLT will need to be determined for each camera. A long wave length camera (7-15 μm) is less susceptible to solar noise and is a better choice for field use.

   a. Connect IR camera to a power source
   b. Check computers for connectivity with the IR camera
   c. Adjust critical settings with the LabVIEW 2010 program DCAL, Version 1.0 (Ryan, 2013)
      i. Frame rate
      ii. Focus

3. Perform dosimetry (Note: This is THE MOST CRITICAL STEP in the entire calibration, and should be done carefully. Any errors in this step systematically propagate throughout the rest of the calibration.) using a WR-10 open ended waveguide antenna connected to an HP/Agilent W8486A power sensor. The power sensor should also be connected to a HP437B power meter. The power sensor and meter are both calibrated to standards from National Institute of Standards and Technology (2013) and the antenna is calibrated against a Millitech Standard Gain Pyramidal Horn (SGH) (ensure calibrations are current). The SGH comes with a frequency versus gain curve from the manufacturer. From this chart (Millitech, Inc., 2013), the effective area of the antenna can be determined.

$$A_e = \frac{\lambda^2}{4\pi} A_g \tag{9}$$

   where A_g is the dB gain from the chart. At 95 GHz, the SGH gain utilized for previous calibrations has been 1.699 cm^2. With A_g determined, a vector network analyzer or signal source and sensor can be utilized to characterize the waveguide and attenuators. Assuming all values are converted to dB, the total loss of the system can then be

determined and programmed into the power meter as an offset to allow for direct readout of the power density. A modification to this process may be required when the field is too small for the SGH. In this occurrence, an Open Ended Waveguide (OEWG) can be utilized; however, there are no gain charts currently provided for this waveguide, so a closely related calibration will be required. This process involves using the SGH to measure the field and then the OEWG is substituted to get the equivalent reading. With this done, the A_g for the OEWG can be calculated. A typical value for the OEWG utilized in previous calibrations is 0.0502 cm^2. This value is dependent on the material, the waveguide length, and the incident angle. Finally, the power sensor is ready to be placed in the plane of exposure. Use a robotic arm to sweep the antenna and power sensor through multiple locations to find the highest power density of the beam (center point). Once the highest power density is achieved, the power density is then divided by the pulse repetition frequency (PRF) to determine the normalized power density of a single pulse. The following steps depict the measurement process after calibrating the equipment.

   a. Position antenna and power sensor in the plane of exposure
   b. Manipulate the antenna until it is aligned with the center of the beam
   c. Divide the highest power density by the PRF to attain the normalized power density
   d. Adjust transmitter PRF for the desired power density

4. Start CLT calibration and control program DCAL
   a. Adjust exposure parameters:
      i. Normalized power density
      ii. Actual power density
      iii. Exposure duration
   b. Adjust IR camera settings:
      i. Ambient temperature
      ii. Emissivity
      iii. Atmospheric temperature
      iv. Relative humidity

5. Place CLT in the proper position
   a. Verify distance is consistent with the distance of the power density measurement and verify CLT is normal to the beam

6. Perform test shot
   a. Verify correct settings and system operation

7. Annotate CLT spot locations in laboratory record (Figure 3). These eight locations are suggested:
   a. Lower left (LL)
   b. Lower center (LC)
   c. Lower right (LR)
   d. Center right (CR)
   e. Upper right (UR)

f. Upper center (UC)
g. Upper left (UL)
h. Center left (CL)

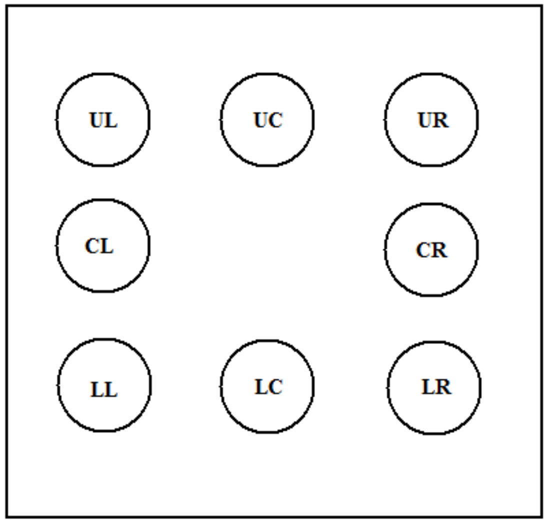


Figure 3. CLT spot locations.

8. Expose CLT locations

9. For each spot record the following data using DCAL (the operation of DCAL is described in Appendix A):
    a. Time
    b. Maximum temperature array
    c. Overall maximum temperature
    d. Array size

4.0 REDUCTION OF DATA

The raw IR data is in the form of time–temperature pairs. The time base is quite variable as to start time. An example is shown in Figure 2. In order to analyze the data and generate a fit to the equation, the actual start time and temperature must be estimated so that it can be subtracted out of the problem. Currently, that is performed automatically with the FORTRAN 90 program called DCALREADER (Beason, 2013). A listing of the program is in Appendix B.

DCALREADER reads all the IR data temperature files, attempts to find the zero time points, and creates two files. One file (paths // outfile // '-raw.csv') contains all the raw data, and the other (paths // outfile // '-zeroed.csv ') contains the all the reduced data.

In order to run DCALREADER, an input file must be created. The input file contains the path to and names of the raw data files, the number of raw data files, and the averaging lengths for the zero location algorithm. A copy of a sample input file for DCALREADER is in Appendix C.

DCALREADER should execute on most Linux/Unix/OSX systems when compiled with the gcc FORTRAN compiler GFORTRAN. One compiles it using the statement:

```
gfortran dcalreader.f90 –Wall
```

and execute it with

```
./a.out < inputfile
```

where a.out is the default name of the executable output from GFORTRAN and inputfile is the input file created. A sample is in Appendix C.

There can be an issue with DCAL in which the number of data points printed out are not the same as the number of data points that are actually in the file. If that occurs, edit the errant data file to reduce the number of data points by 1, and try again.

Shown in Figure 4 is a plot of a subset of the DCALREADER output from the 10 s, 0.5 W-cm^{-2} shots for the CLT012 calibration. There are three possible reasons for the differences seen in the slopes and final temperatures seen between the eight shots that were taken.

The first possibility is a shot-to-shot variation in the output of the source. The source output variation would be random.

The second possibility involves the fact that each of the shots shown was acquired at a different location on the CLT, thus inhomogeneities of the carbon mix could account for variation. This variation would be systematic, and should be repeatable if the exact spot was shot again.

The third possibility involves the angle at which the beam strikes the CLT. Beam angles that are not normal to the CLT surface create greater reflections and less heating of the CLT. This variation would be random provided that the method for mounting the CLT was not exactly repeatable.

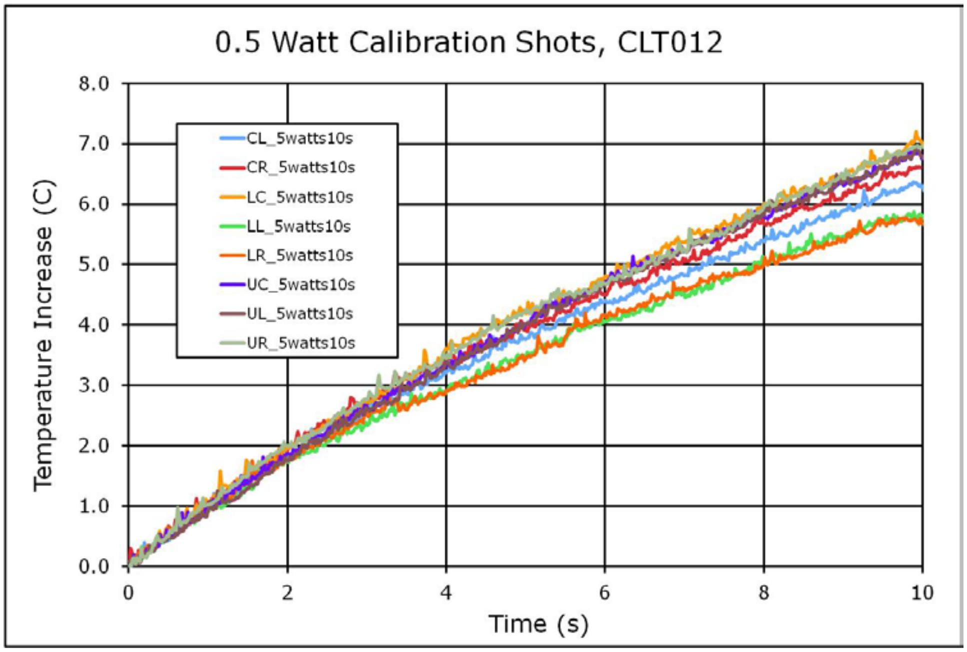


Figure 4. Results from 10 s, 0.5 W-cm^{-2} shots on CLT012.

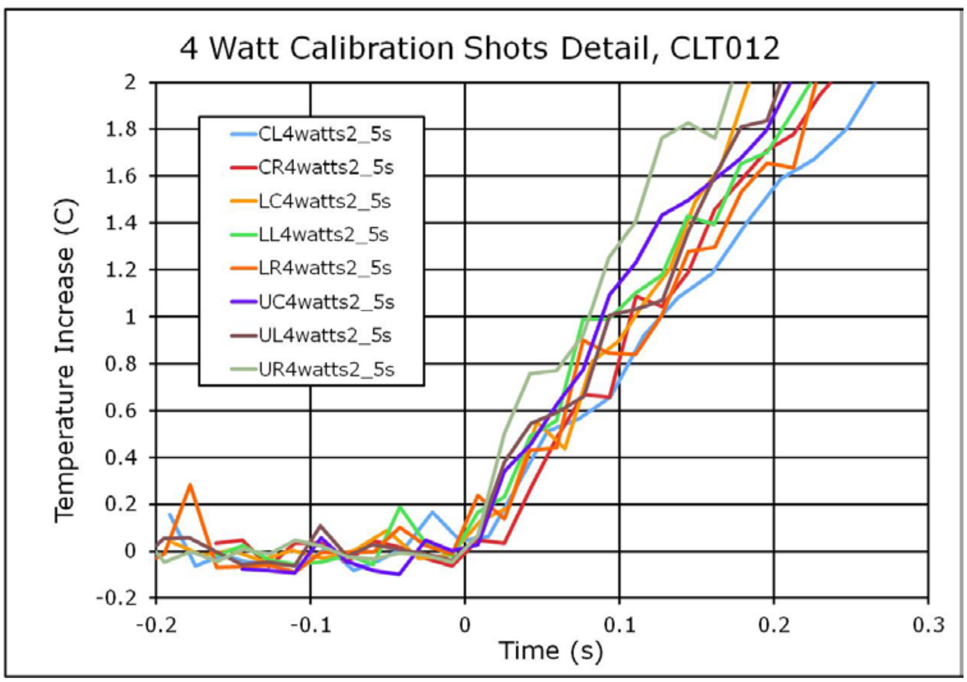


Figure 5. Results detail from 2.5 s, 4 W-cm^{-2} shots on CLT012.

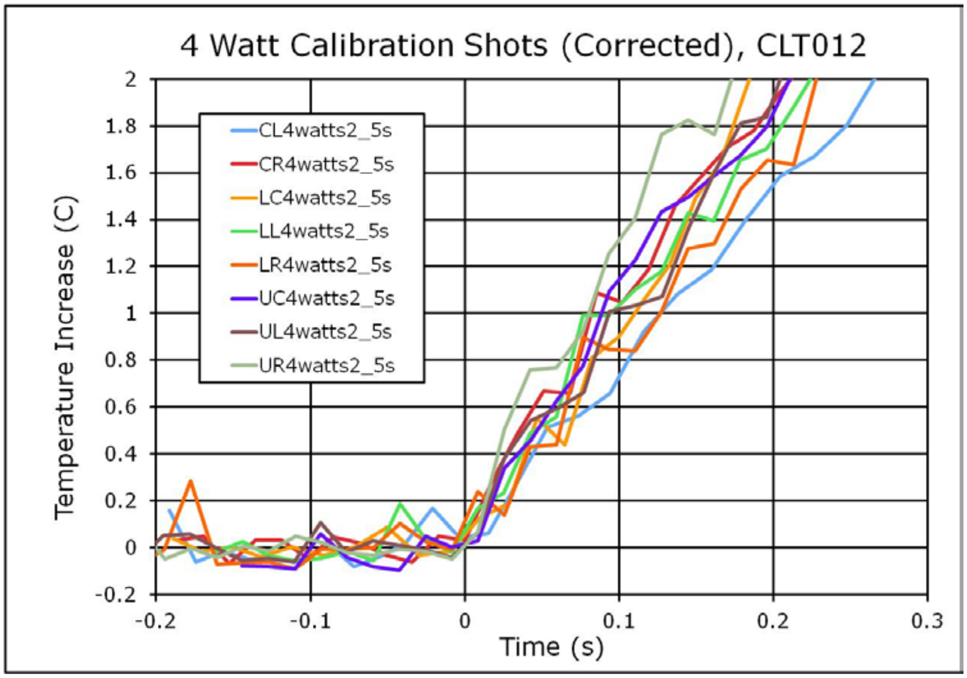


Figure 6. Results detail from 2.5 s, 4 W-cm^{-2} shots on CLT012, time base corrected.

An issue with the zero time finding algorithm in DCALREADER is that it does not always find an acceptable zero time. This issue is illustrated in Figure 5 and Figure 7. The offending shot designators are CR4watts2_5s for Figure 5 and UC6watts2s, LR6watts2s, and UR6watts2s for Figure 7.

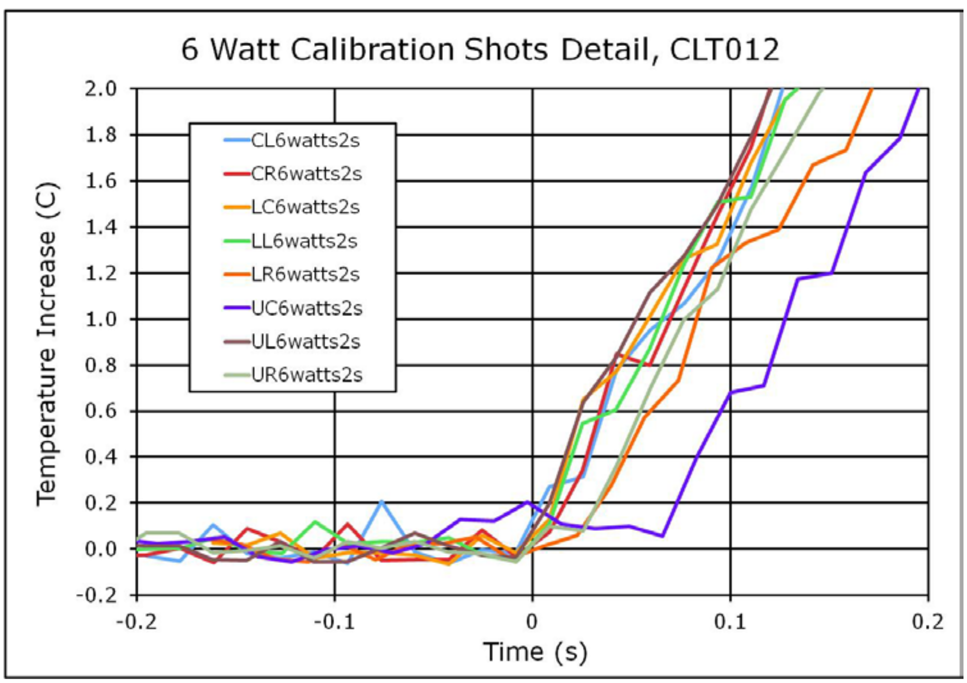


Figure 7. Results detail from 2 s, 6 W-cm^{-2} shots on CLT012s.

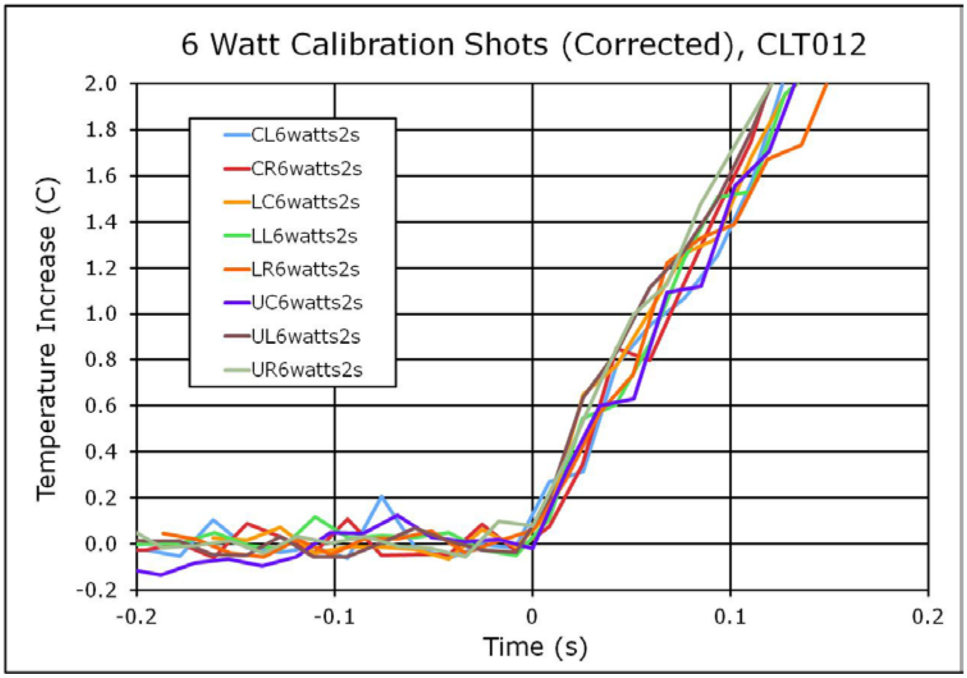


Figure 8. Results detail from 2 s, 6 W-cm^{-2} shots on CLT012, time base corrected.

These data are hand corrected, with the results shown in Figure 6 and Figure 8. All of the calibration data shot results are shown in Appendix D.

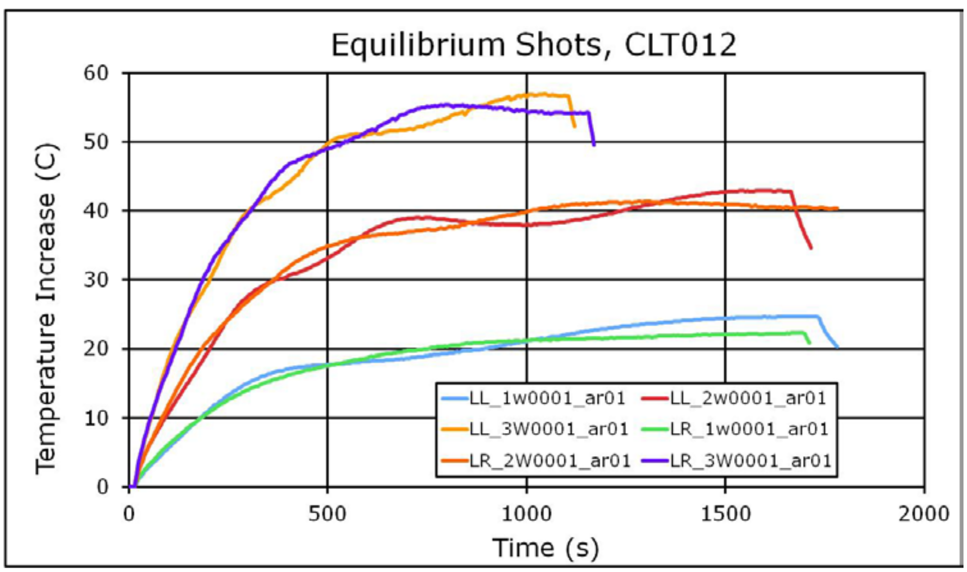


Figure 9. Results from the equilibrium shots on CLT012.

In Figure 9, the results of the six equilibrium shots for CLT012 are shown. The variations seen in the pulses are believed to be related to the source power drifting, or the heating, ventilation, and air conditioning system cycling on and off.

In order to determine the equilibrium temperature versus power, the power for the same power shots are averaged to create the data shown in Figure 10.

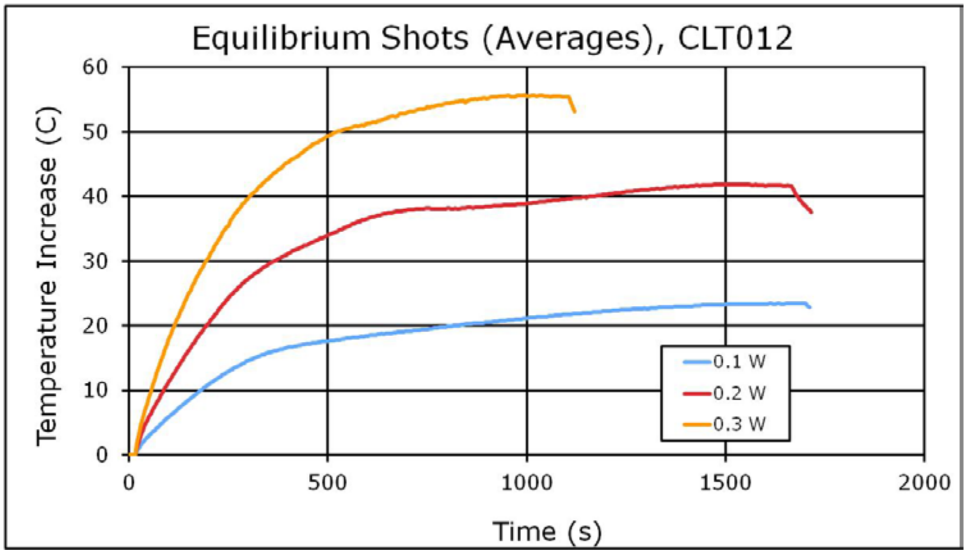


Figure 10. Averaged results from the equilibrium shots on CLT012.

The last 20 time data points before the source was turned off are averaged for each power. These data are shown as the three data points in Figure 11. A linear regression with the intercept set to zero was then used to generate a linear fit through the three data points. This fit is shown in Figure 12.

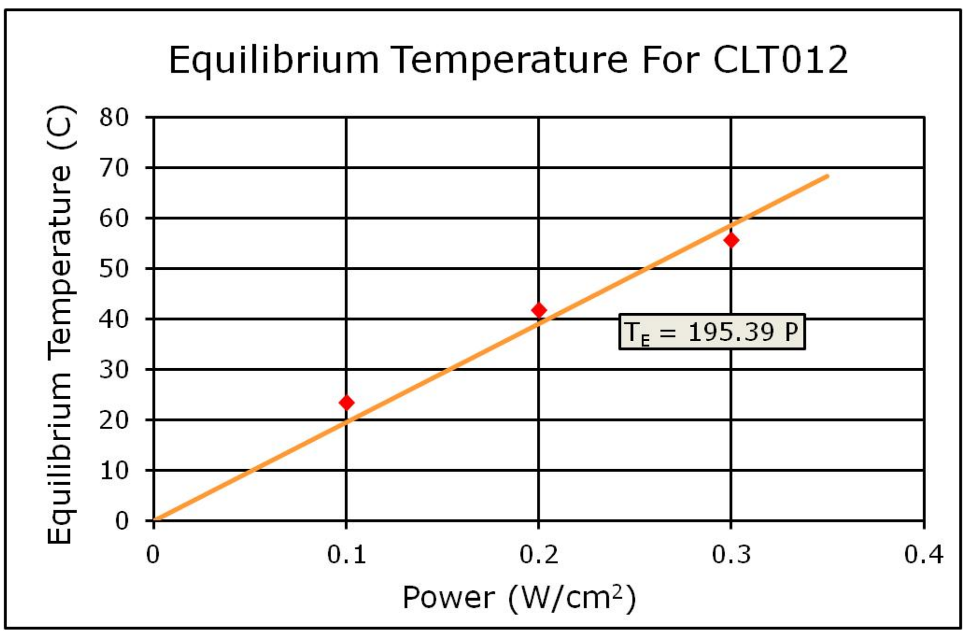


Figure 11. Equilibrium temperature fit for CLT012.

5.0 ANALYSIS OF DATA

The data from DCALREADER is used in a *Mathematica®* notebook that uses the *Mathematica®* function *NonlinearRegress*. A truncated form of the notebook used is shown in Appendix E. The truncation was accomplished by simply removing the majority of the data time/power/temperature data used for the fit from the printout. For this example, *Mathematica®* (Version 5.2) was used. The entire dataset for this calibration contains 13,820 data triplets which would require 400 pages. The authors can be contacted for complete dataset if desired.

In the notebook, the array called data contains the {time , power , temperature} triplets obtained from the calibration data acquisition phase. The format is:

```
data = { { time1 , power1 , temp1 } , { time2 , power2 , temp2 } ,
         { time3 , power3 , temp3 } , { time4 , power4 , temp4 } ,
           ... ,
         { timeN , powerN , tempN } }
```

It is possible (and likely) there will be multiple {time , power , temperature} triplets that are the same that come from different CLT exposures during the calibration data. There is no issue with this approach, and it is important that the repeats are not removed. This is because the repeated data will provide more heavily weighted data points for the non-linear regression performed by *Mathematica®*.

The *Mathematica®* notebook (see Appendix E, page 50) shows the best fit parameters that were found for fitting Equation 8, and are listed in the printout as “b” (for β), “c” (for γ), and “e” (for ϵ). The values found by the fit are $\beta = 1.14964$, $\gamma = 0.197516$, and $\epsilon = 0.00615329$.

The fit obtained is shown graphically in 3-D (Figure 12).

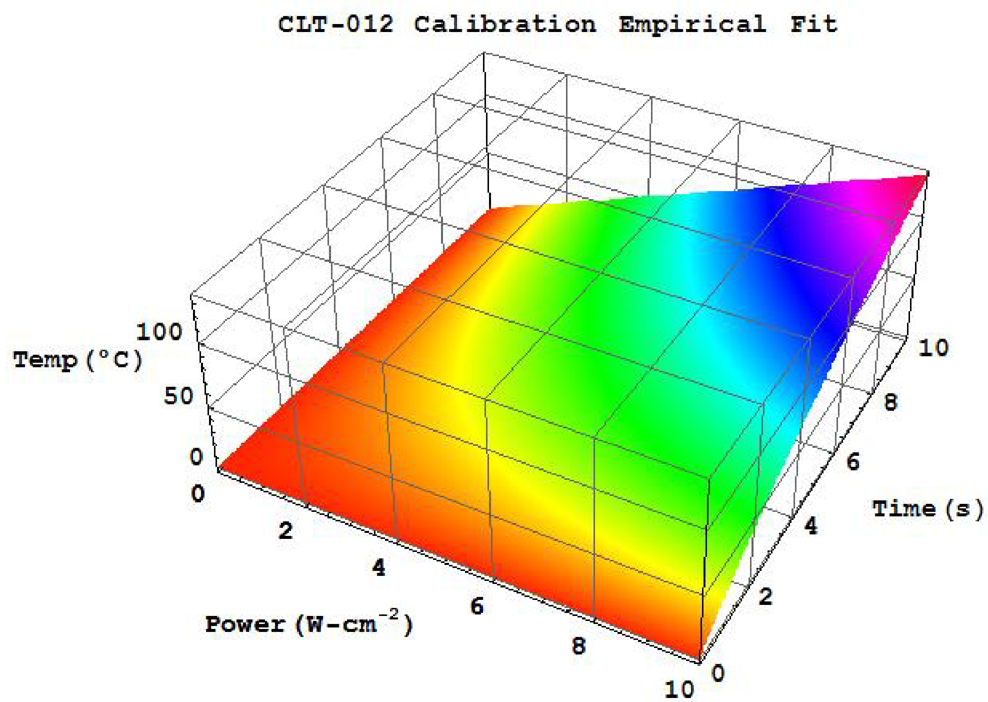


Figure 12. Calibration fit result from *Mathematica*® for CLT012.

A comparison of the fit with the calibration data is shown below in Figures 13-17.

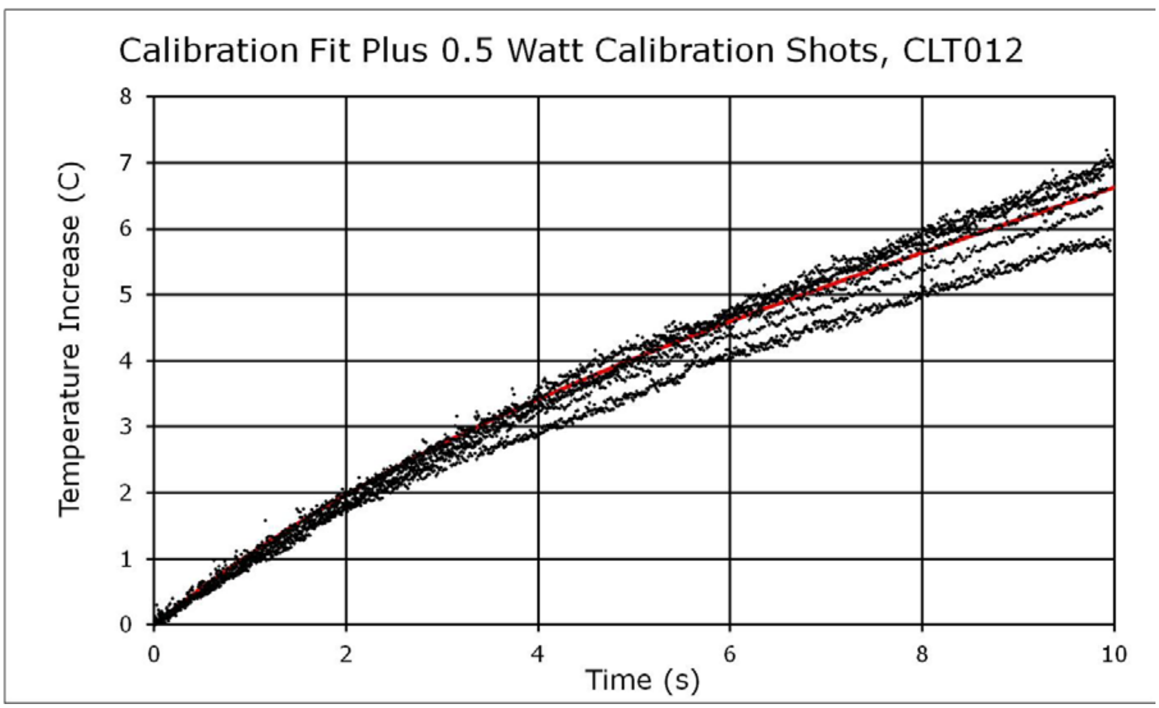


Figure 13. Comparison of fit to data, 0.5 W.

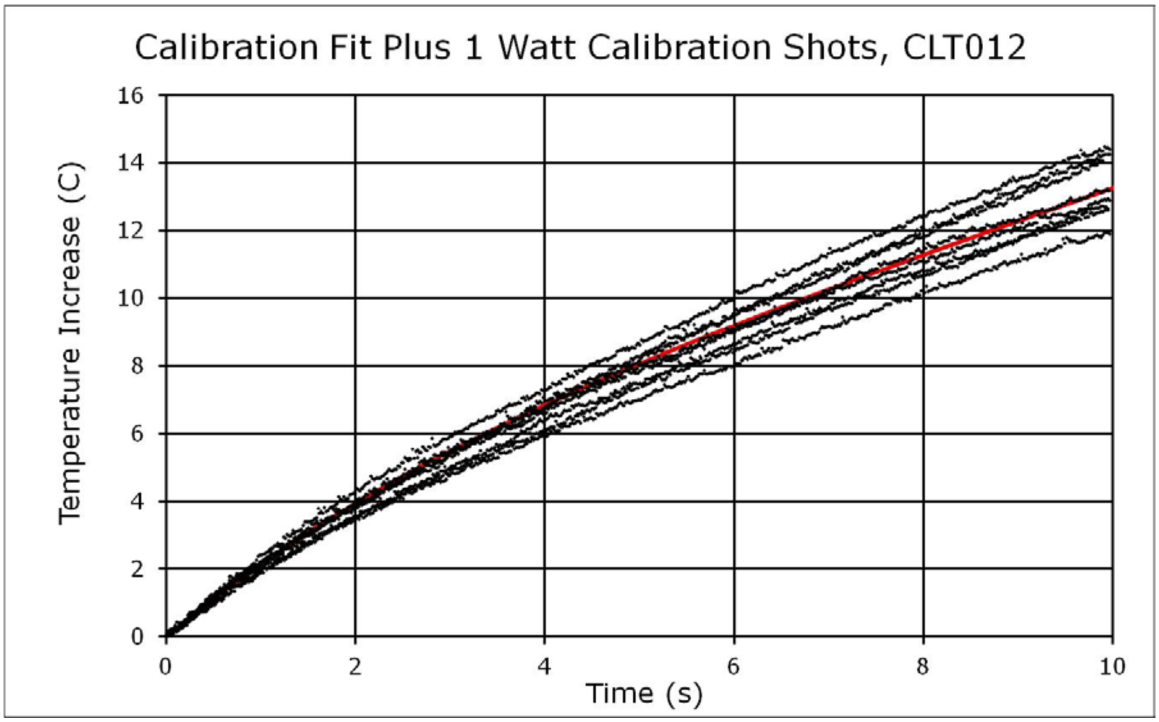


Figure 14. Comparison of fit to data, 1 W.

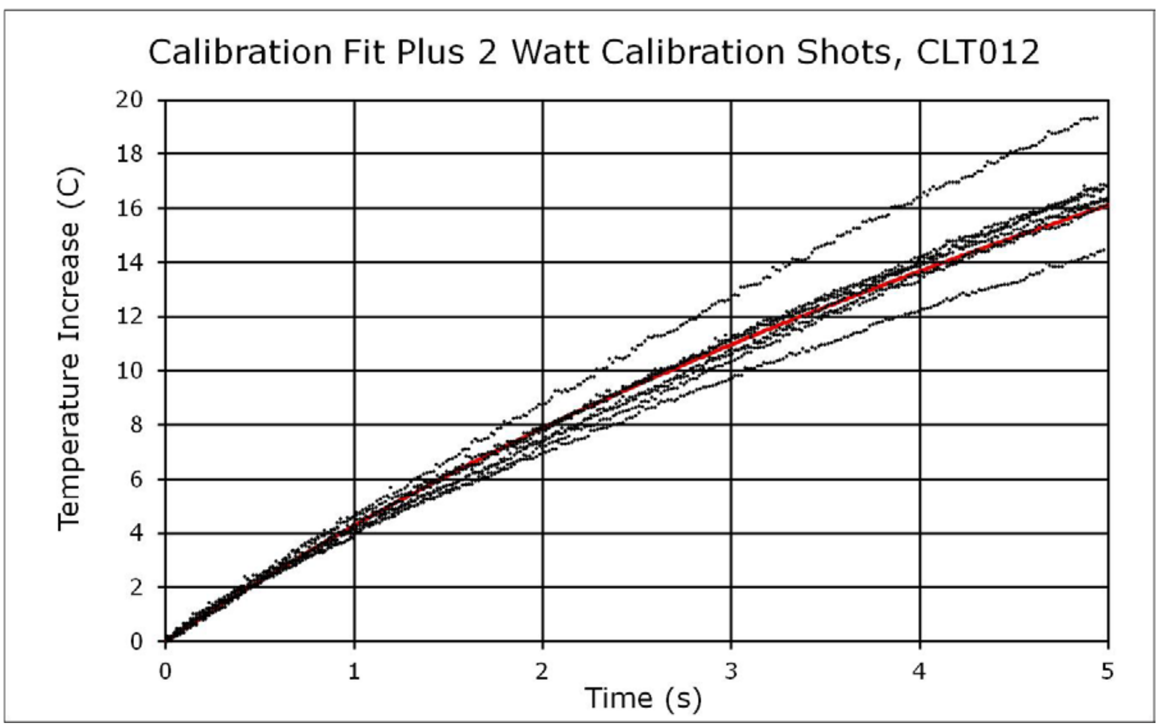


Figure 15. Comparison of fit to data, 2 W.

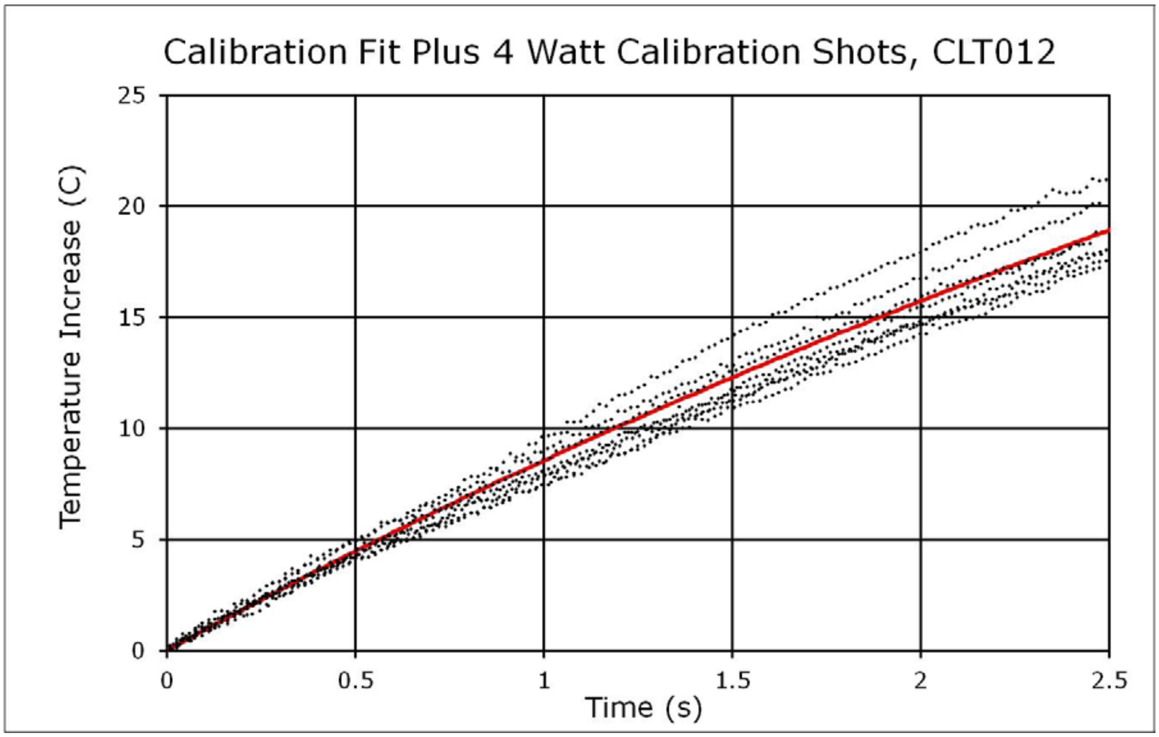


Figure 16. Comparison of fit to data, 4 W.

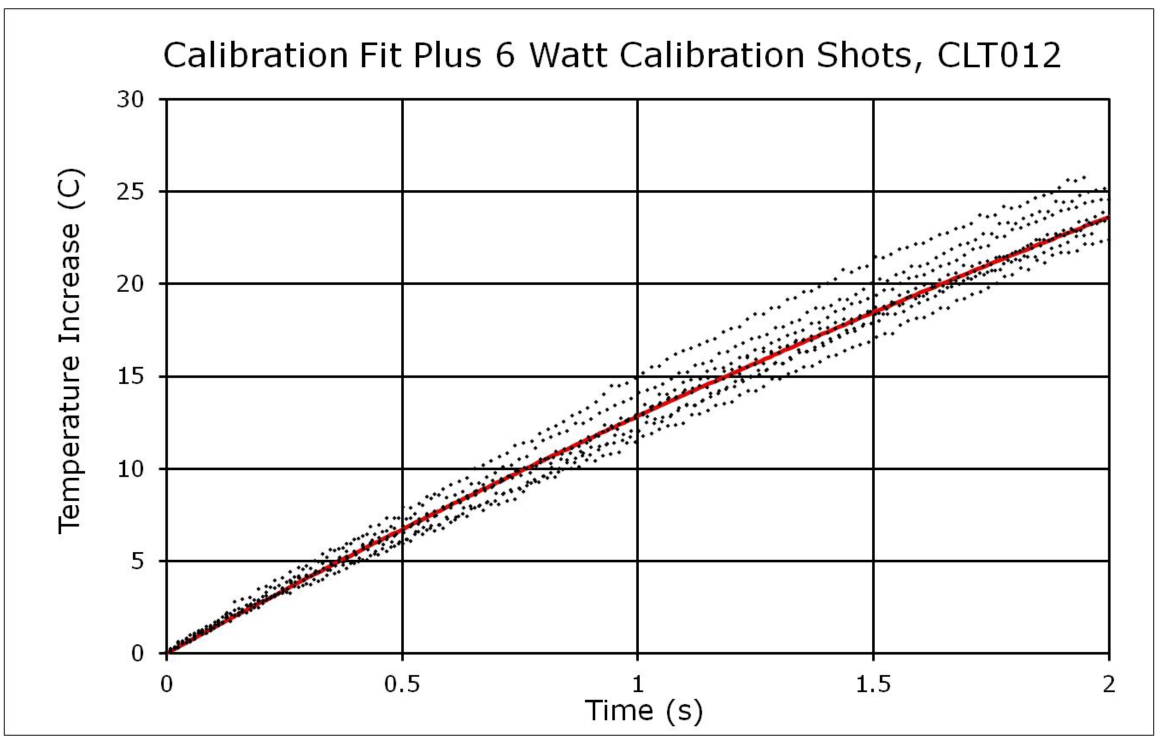


Figure 17. Comparison of fit to data, 6 W.

6.0 CONCLUSION

A methodology for calibrating an areal calorimetry-based dosimeter has been presented. The methodology presented uses CLT for the calibration example. However, the same methodology can be applicable to thermal calibration of other sheet target materials.

The code for all data analysis tools used in the calibration (Appendices A, B, C, and E) has been included.

REFERENCES

Beason, C. W. (2005). *CLT As A Calorimeter.* PowerPoint presentation dated 10 January 2005, presented to AFRL/RHDA at HERTF on 10 January 2005, 37 pp, FOUO.

Beason, C. W. (2013, April). DCALREADER(Version 0.03)[computer program]. JBSA Fort Sam Houston, Texas: General Dynamics Information Technology.

Millitech, Inc. (2013). *Series SGH Standard Gain Conical and Pyramidal Horns.* Retrieved April 30, 2013, from http://www.millitech.com/pdfs/specsheets/IS000025-SGH.pdf

Mylacraine, K. S. (2013). *Standard Operating Procedure for Northstar 94 GHz Radio Frequency Source*. Unpublished Report. Fort Sam Houston, TX: Air Force Research Laboratory, 711th Human Performance Wing, Human Effectiveness Directorate, Bioeffects Division, Radio Frequency Bioeffects Branch.

National Institute of Standards and Technology. (2013, Apr). *Electromagnetic Measurements Power and Energy Measurements, Low-Frequency.* Retrieved April 30, 2013, from http://www.nist.gov/calibrations/power-energy.cfm#56200C

Ryan, D. (2013, April). LabVIEW Program DCAL (Version 1.0) [Computer program]. JBSA Fort Sam Houston, Texas: General Dynamics Information Technology.

APPENDIX A – LABVIEW PROGRAM DCAL

Overview

The LabVIEW program DCAL version 1.0 was developed during November 2012 to control a 94 GHz RF source, and a FLIR IR camera. This program can also be utilized to acquire data from the IR camera for further post processing. To use this software, the user should have LabVIEW 2010 or above. The following figures depict the user interface and block diagrams for the entire program.

DCAL Front Panel

Figure A-1 depicts the main front panel of LabVIEW program DCAL. This front panel provides the user with the ability to control the 94 GHz RF source. It consists of user controls, user indicators, and user input parameters.

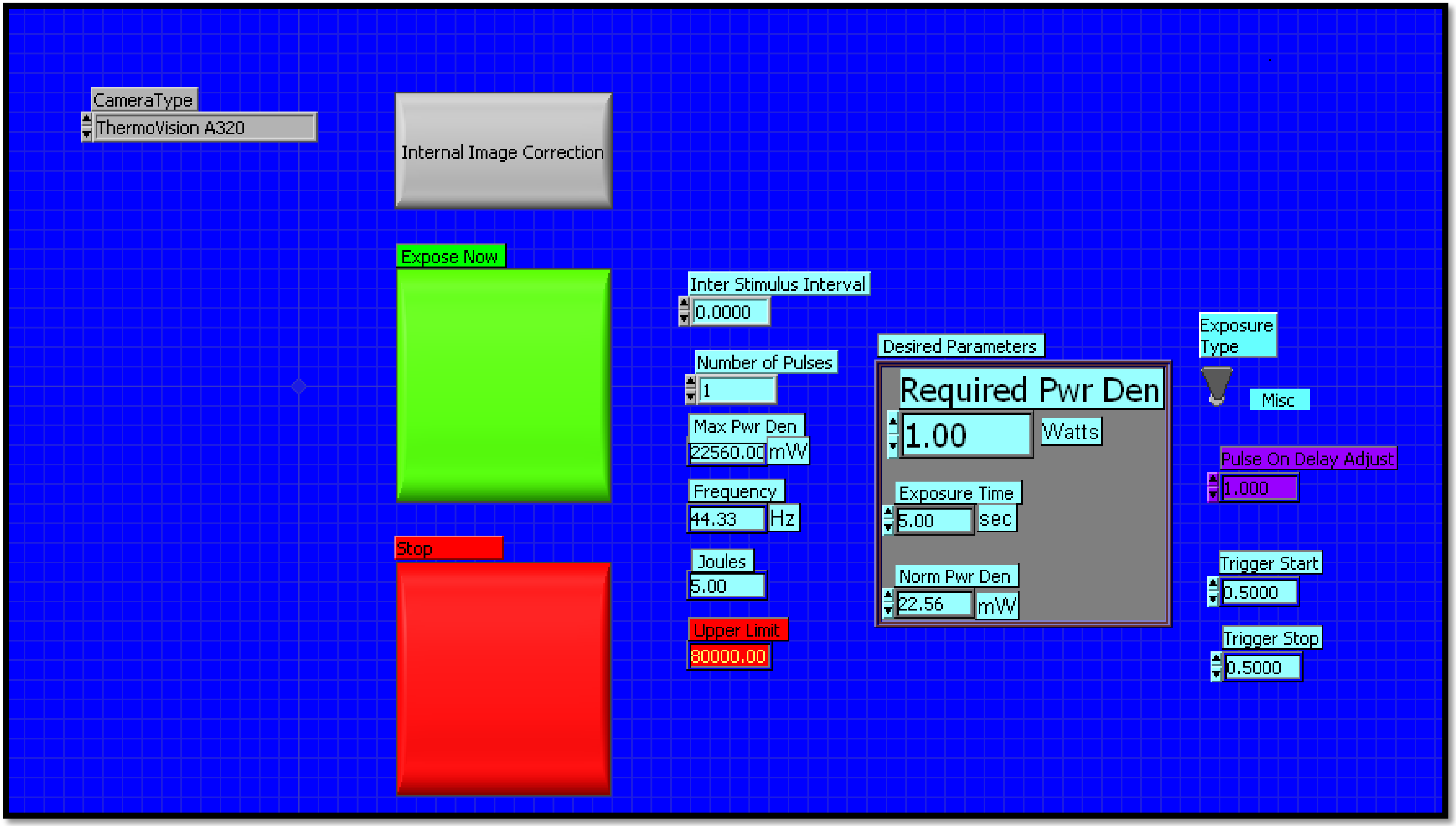


Figure A-1. DCAL Front Panel.

Figure A-2 depicts the user controls for the 94 GHz RF source. The CameraType control is a drop-down menu list of different IR cameras from which the user can choose. The Internal Image Correction button is a function to calibrate the IR camera. This function is a critical feature for controlling the IR camera, and should be utilized before each exposure to maintain measuring accuracy and correct imaging. The Expose Now and Stop button give the user the capability to turn the source on and off.

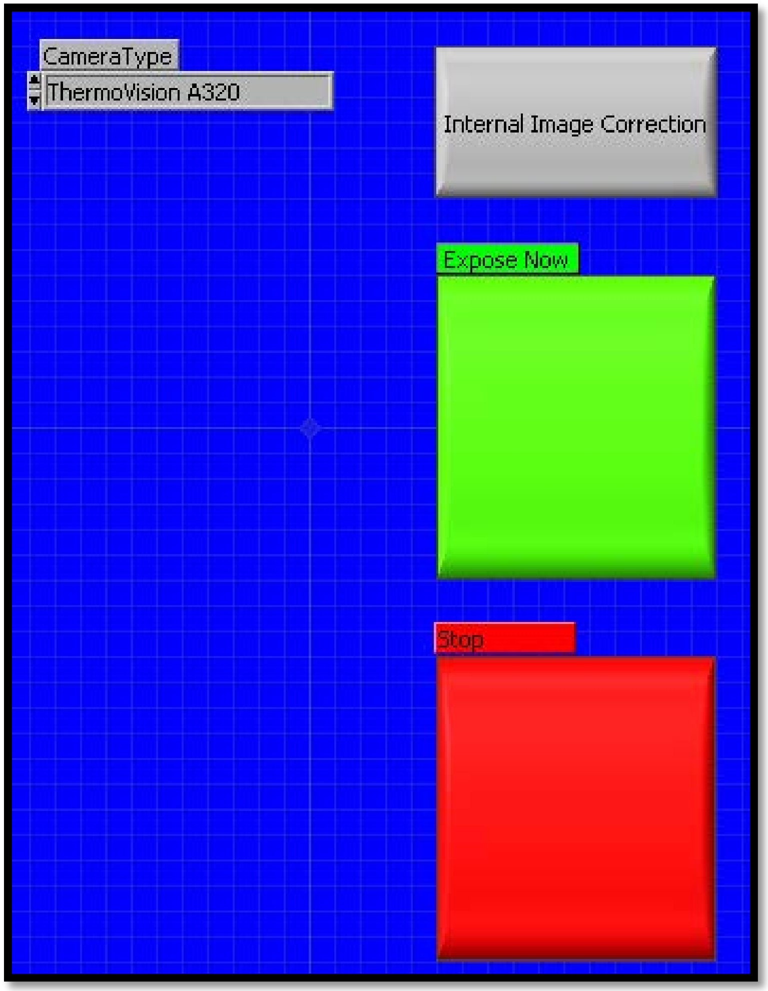


Figure A-2. DCAL Front Panel: User Controls.

Figure A-3 depicts the user input parameters for the 94 GHz RF source. With these parameter settings, the user can control the pulse duration of the exposure, inter stimulus interval, power density, frequency, and the number of pulses.

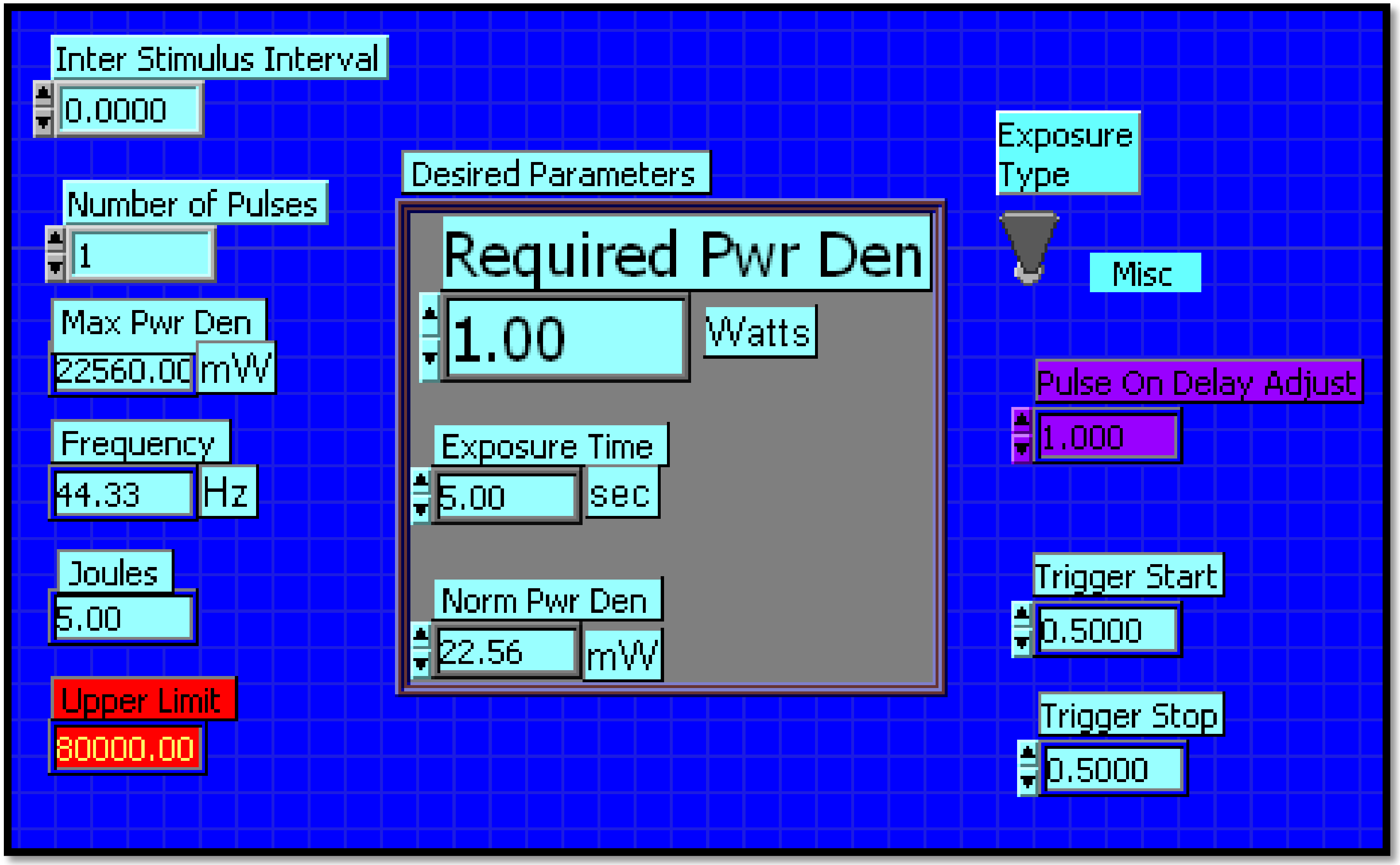


Figure A-3. DCAL Front Panel: User Inputs.

DCAL Block Diagram

The beginning of the block diagram starts with Figure A-4. This portion of the code initializes communication with the source through a General Purpose Interface Bus (GPIB).

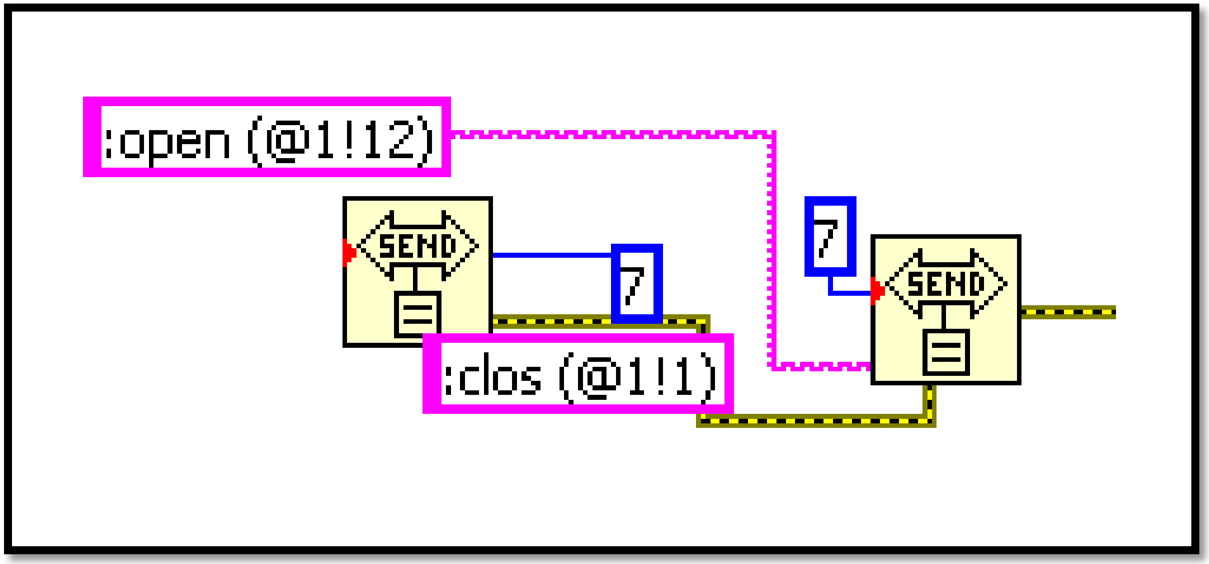


Figure A-4. DCAL Block Diagram: GPIB.

Figure A-5 depicts the portion of the block diagram that controls the IR camera. This portion of the graphical code creates and initializes a ThermoVision™ object. After initializing the object, the subVI "IR Monitor" is then initiated. A subVI is the LabVIEW equivalent to "functions," "subroutines," and "methods" in other programming languages. The purpose of this subVI is for camera control and data acquisition. Once the subVI is complete, the last part of this code destroys the ThermoVision™ object created by the ThermoVision™ Open VI. Please see Figures A-10 through A-15 below for the front panel and block diagram of the IR Monitor subVI.

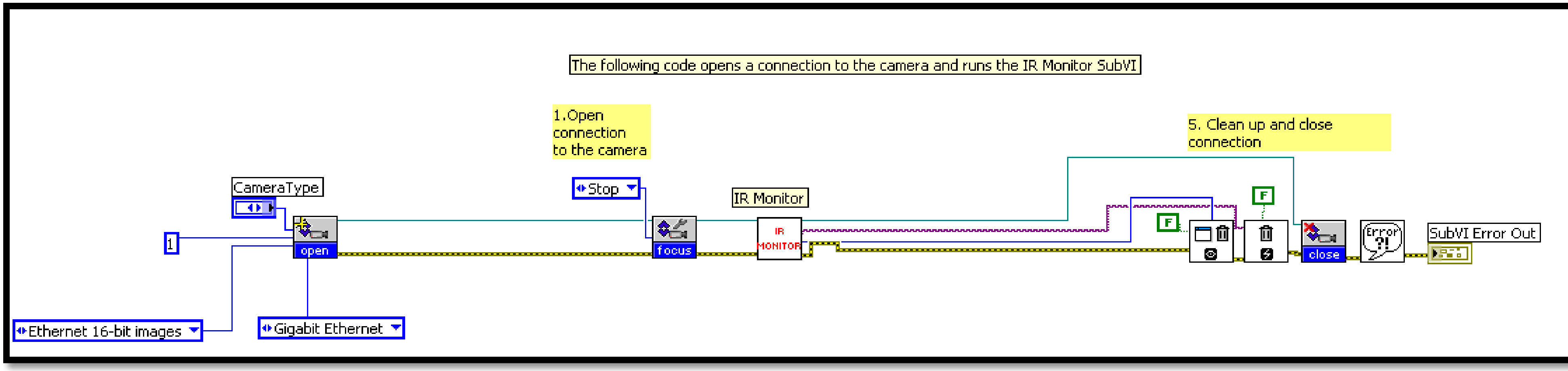


Figure A-5. DCAL Block Diagram: Initializing IR Camera.

Figure A-6 depicts the portion of the block diagram where the user parameters are implemented. This code runs continuously and collects the user parameters until the exposure button is pressed. Once the exposure button is pressed, the parameters set by the user are sent to the source through GPIB command VI's.

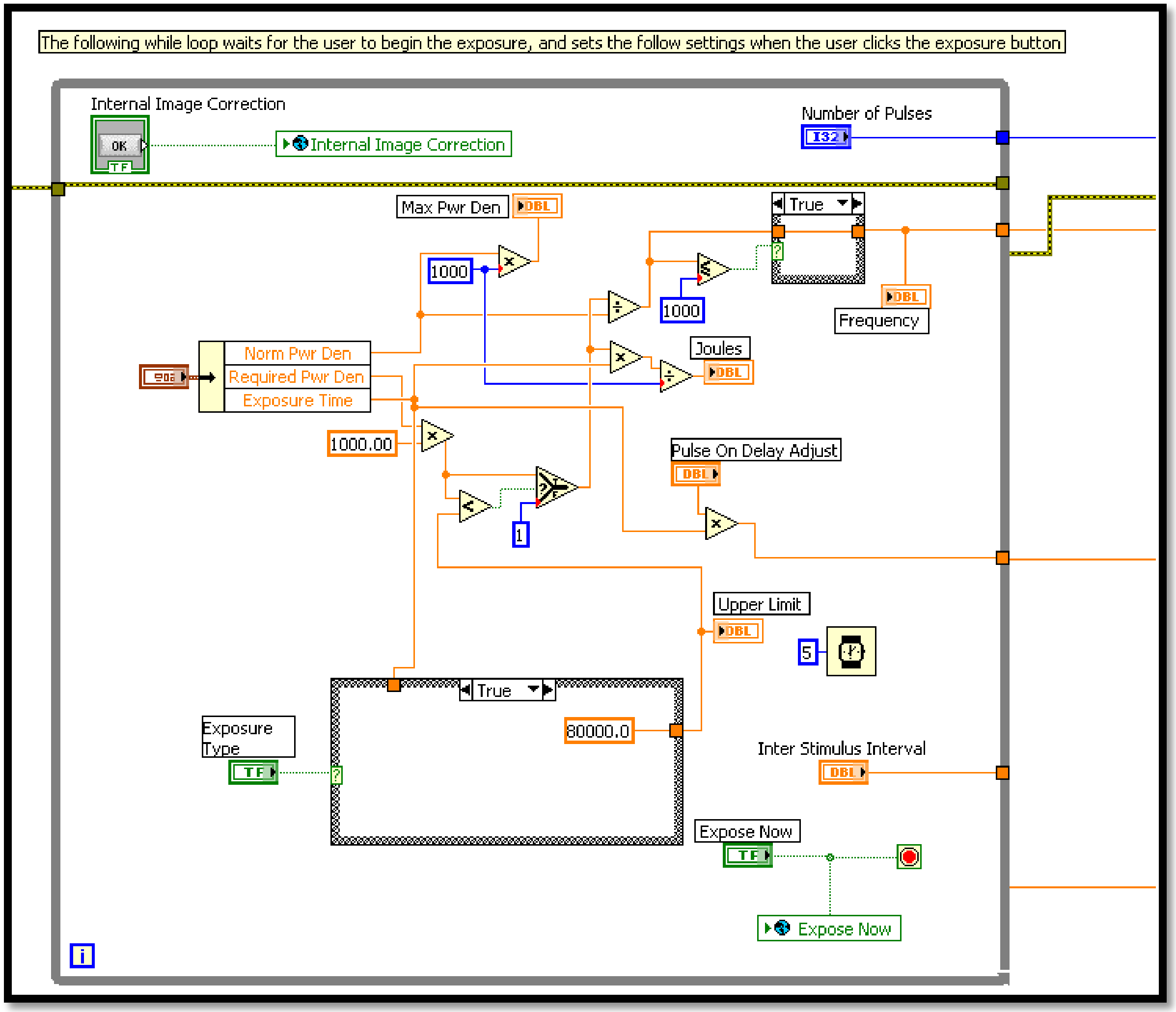


Figure A-6. DCAL Block Diagram: Set User Parameters.

The graphical code below represents the commands sent to the source via GPIB. Starting from left and working right in Figure A-7, the code first sets the frequency for the pulse generator, and then the pulse leading edge. Next, a timer is set based on the input from the user for the pulse duration. The last VI sends a final command to set the start pulse trailing edge for the source.

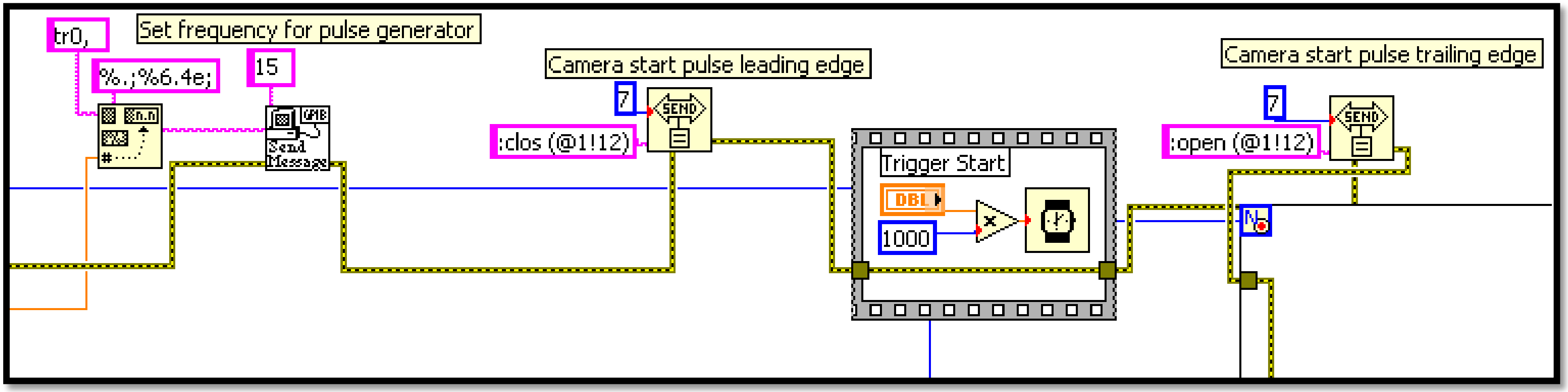


Figure A-7. DCAL Block Diagram: 94 GHz RF Source Control.

Figure A-8 represents additional VI's to communicate with the source. In this section of the block diagram, a command is sent to the source to switch a light on. This light on the source indicates to the user that the transmitter has been initiated. The next command switches on the transmitter. Finally, an event structure is implemented to switch the transmitter off when the user selects the stop button on the front panel.

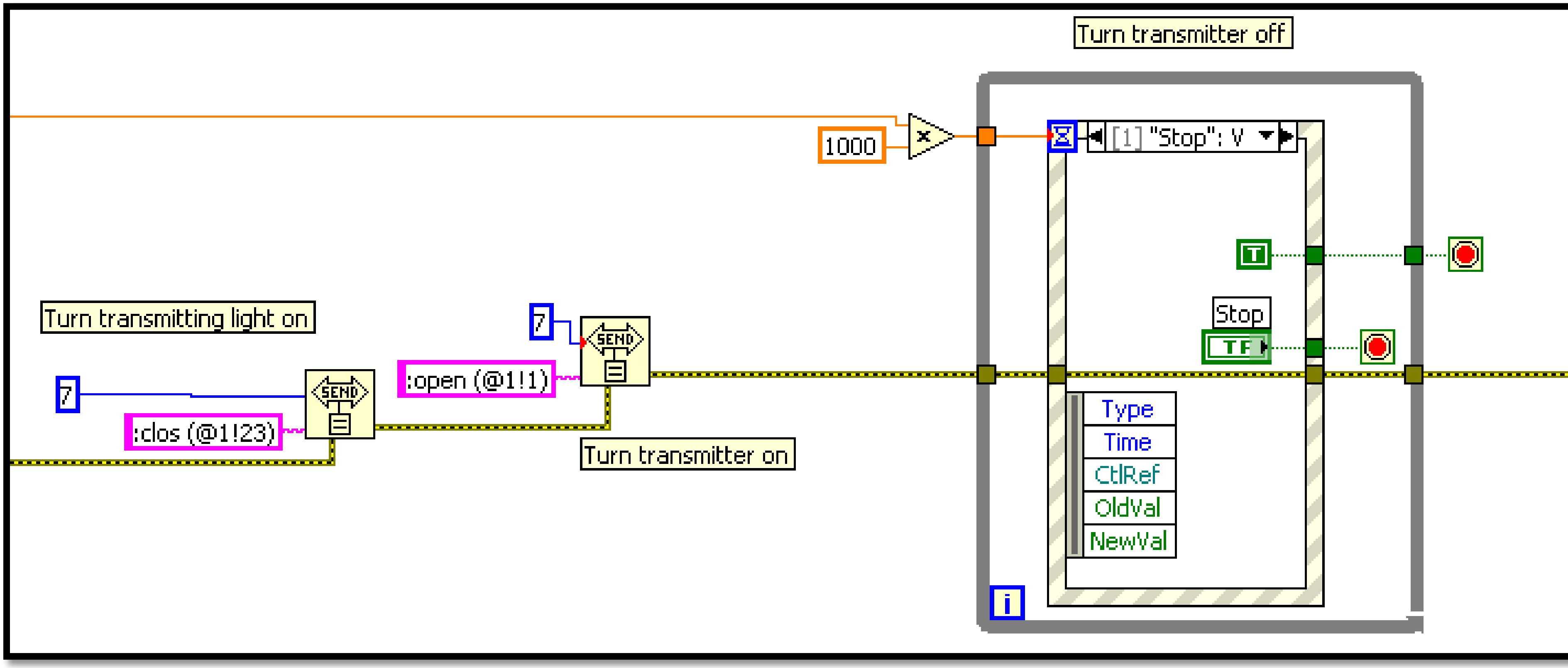


Figure A-8. DCAL Block Diagram: 94 GHz RF Source Control.

Once the pulse duration has been reached, a command is sent to switch the light on the source off, and to stop transmitting altogether. The last part of the code shows a millisecond timer for inter-stimulus interval functionality.

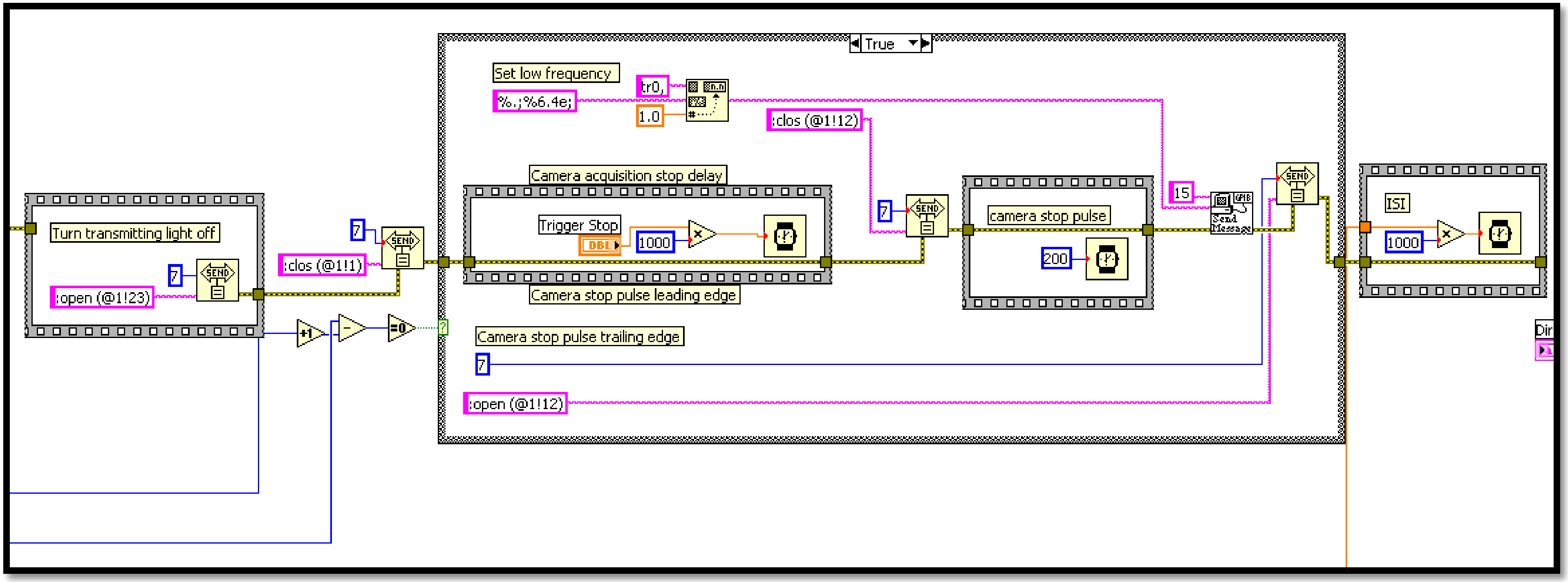


Figure A-9. DCAL Block Diagram: 94 GHz RF Source Control.

Distribution A: Approved for public release; distribution unlimited (P.A. Case No. TSRL-PA-2015-0136, 24 Sep 15).

IR Monitor Front Panel

Figure A-10 depicts the front panel for the IR Monitor subVI. This front panel is a user interface designed to give the user the ability to set the IR camera parameters, and acquire data.

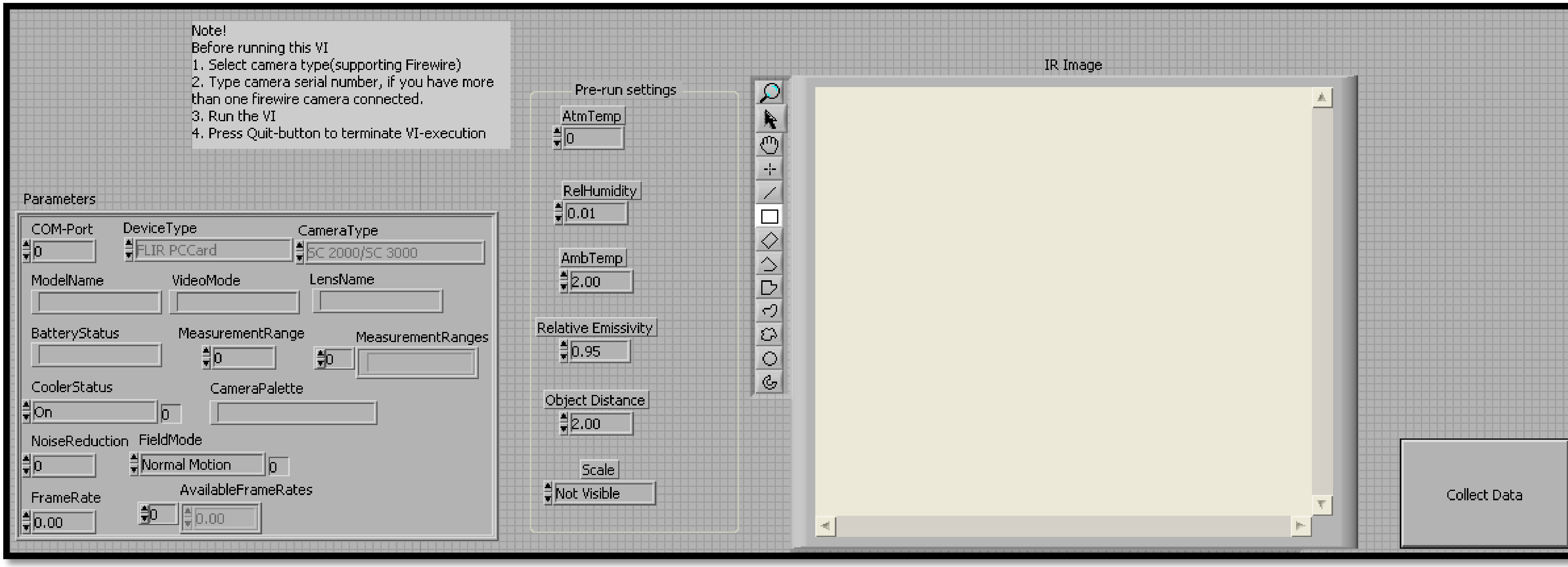


Figure A-10. IR Monitor Front Panel.

Figure A-11 displays the portion of the front panel where the user can input the desired camera settings. The most critical settings to input are the frame rate, atmospheric temperature, relative humidity, ambient temperature, relative emissivity, and object distance.

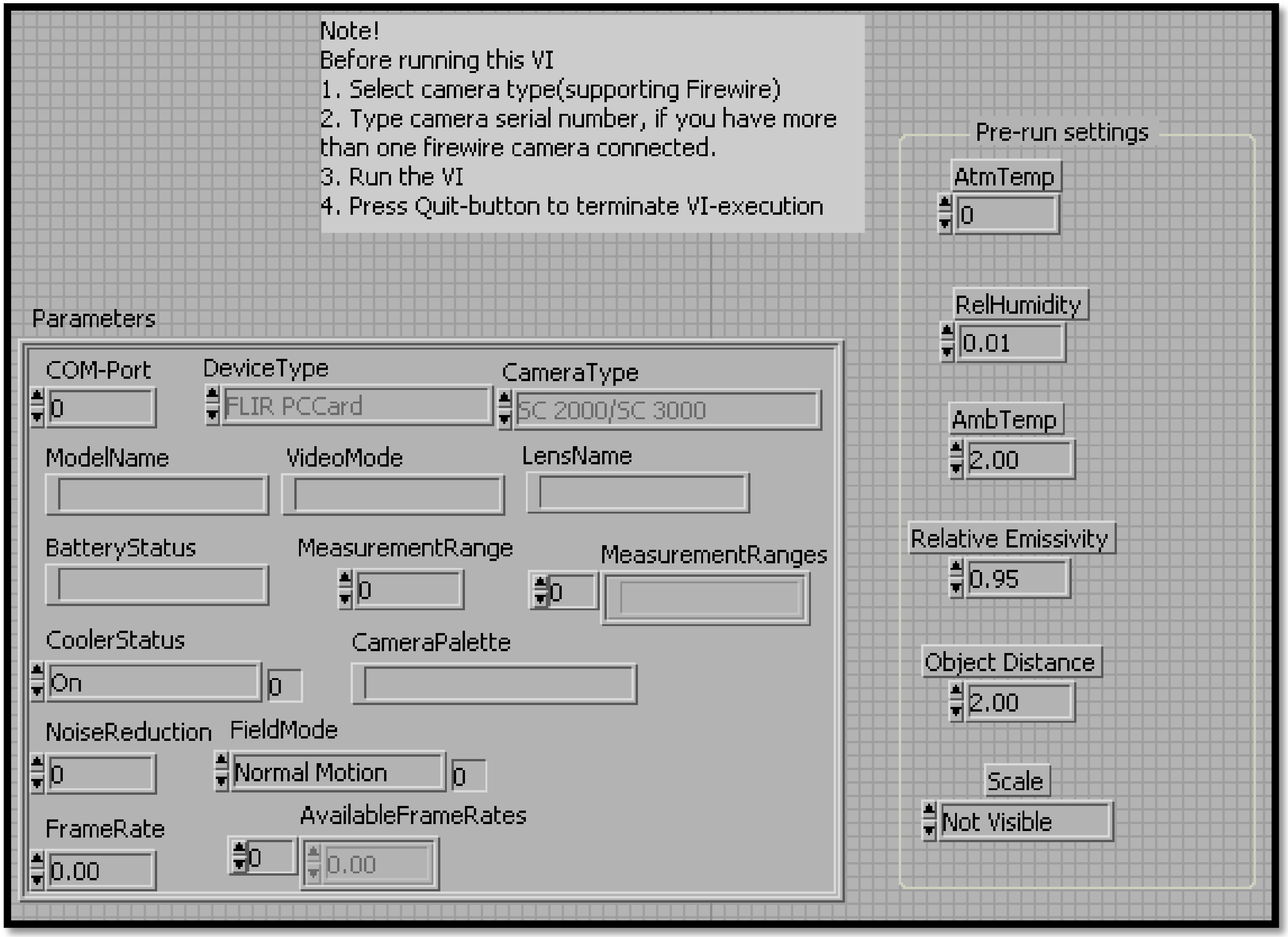


Figure A-11. IR Monitor Front Panel: Camera Settings.

Figure A-12 depicts the portion of the front panel that displays the image to the user. With this feature, the user can visually check for camera connectivity, and verify that the image settings are accurate. The user is also provided tools on the side of the IR image to zoom in and create regions of interests (ROI).

Figure A-12. IR Monitor Front Panel: IR Image.

Figure A-13 displays the user control button to collect data. When this button is pressed, the data from the camera will be stored as a spreadsheet on the computer.

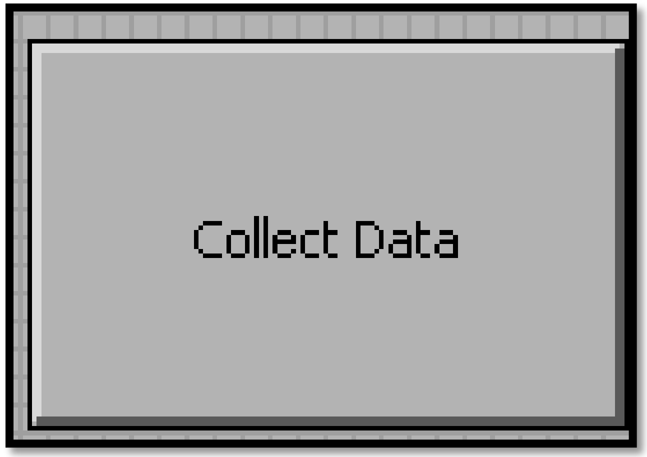


Figure A-13. IR Monitor Front Panel: User Button.

IR Monitor Block Diagram

Figure A-14 depicts the portion of the block diagram that implements the object parameters set by the user. The next part of this code allocates memory for the image and prepares the image to be displayed to the user.

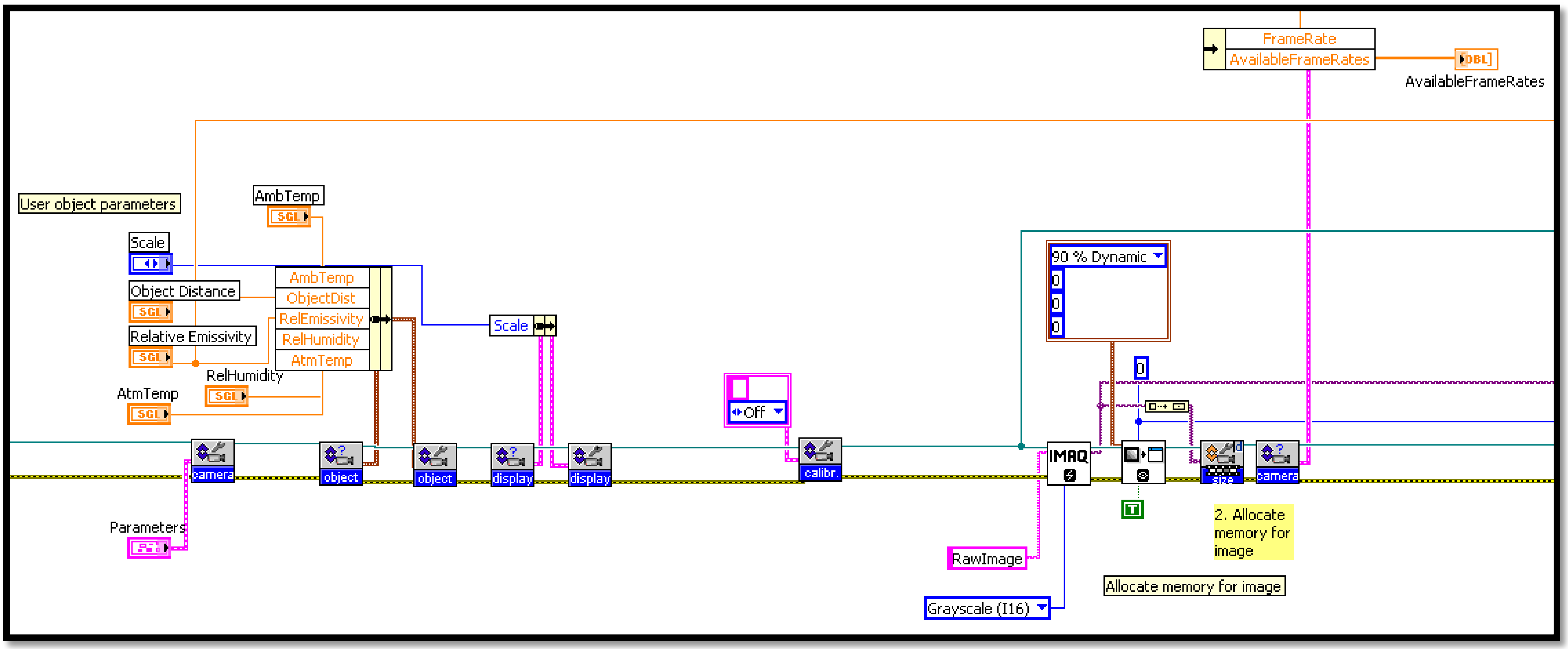


Figure A-14. IR Monitor Block Diagram: Set Camera Object Parameters.

Figure A-15 depicts the portion of the block diagram that starts, requires, and releases the sequence acquisition. The image is captured frame by frame and displayed to the user. Then, the maximum temperature pixel is extracted from the image, and is stored in an array for further processing which takes place in subsequent code.

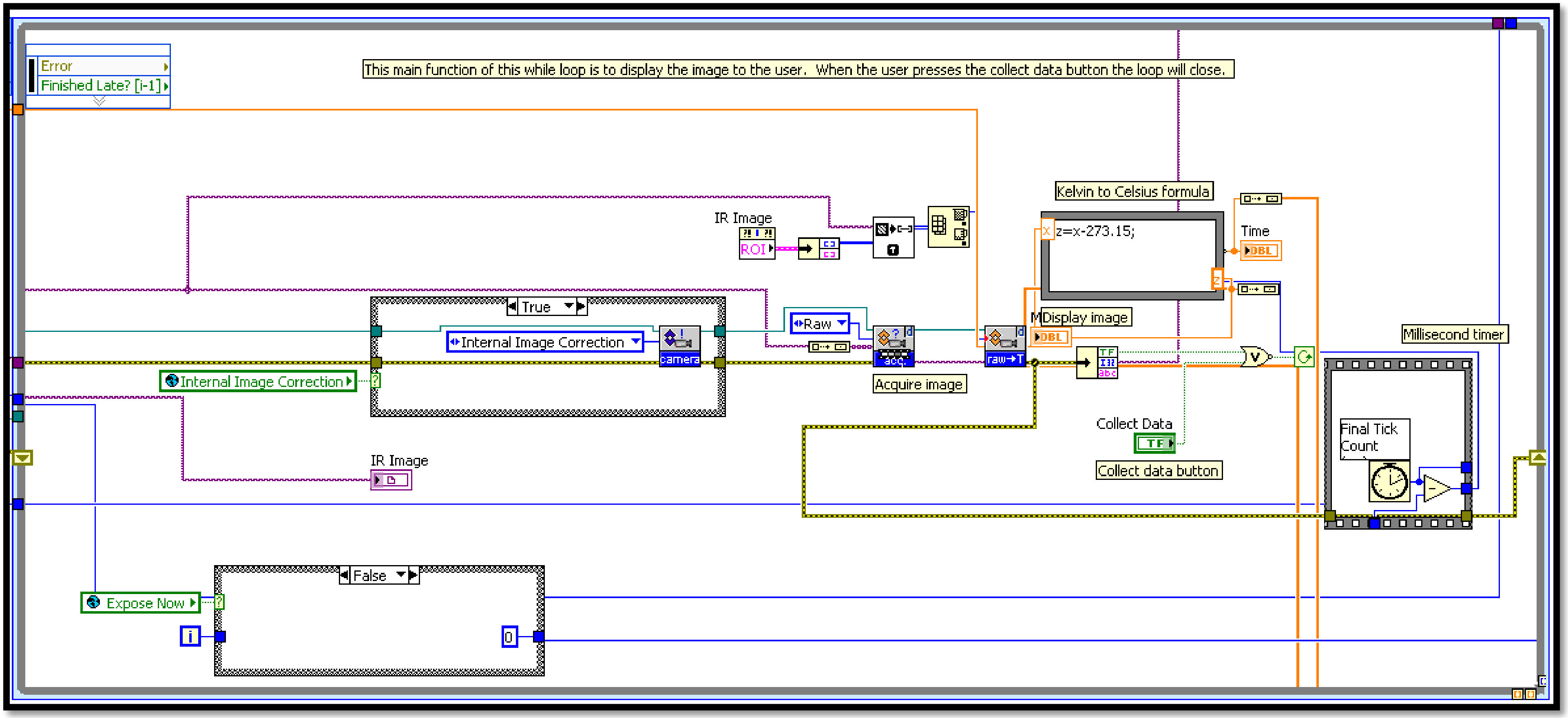


Figure A-15. IR Monitor Block Diagram: Display IR Image.

Distribution A: Approved for public release; distribution unlimited (P.A. Case No. TSRL-PA-2015-0136, 24 Sep 15).

The next critical portion of the block diagram processes the data and prepares it for further analysis by DCALREADER (Figure A-16). Before the data is stored, it is important to know the vicinity of the start time or the frame at which the source was switched on. The algorithm below illustrates this process.

1. Take the average of the first ten temperature pixels after the exposure button is pressed by the user.
2. Subtract each temperature pixel from the average and compare if the difference is greater than or equal to zero.
3. Initialize an array of seven false boolean values. If the difference is greater than or equal to zero, store a true boolean value in the seven-element array; if the difference is less than zero, store a false boolean value in the array.
4. Search the one-dimensional array for false values, if any element is false repeat the loop; if all elements in the array are true, then stop the loop and store that index of the temperature pixel array.
5. Truncate the temperature pixel array starting at this index.

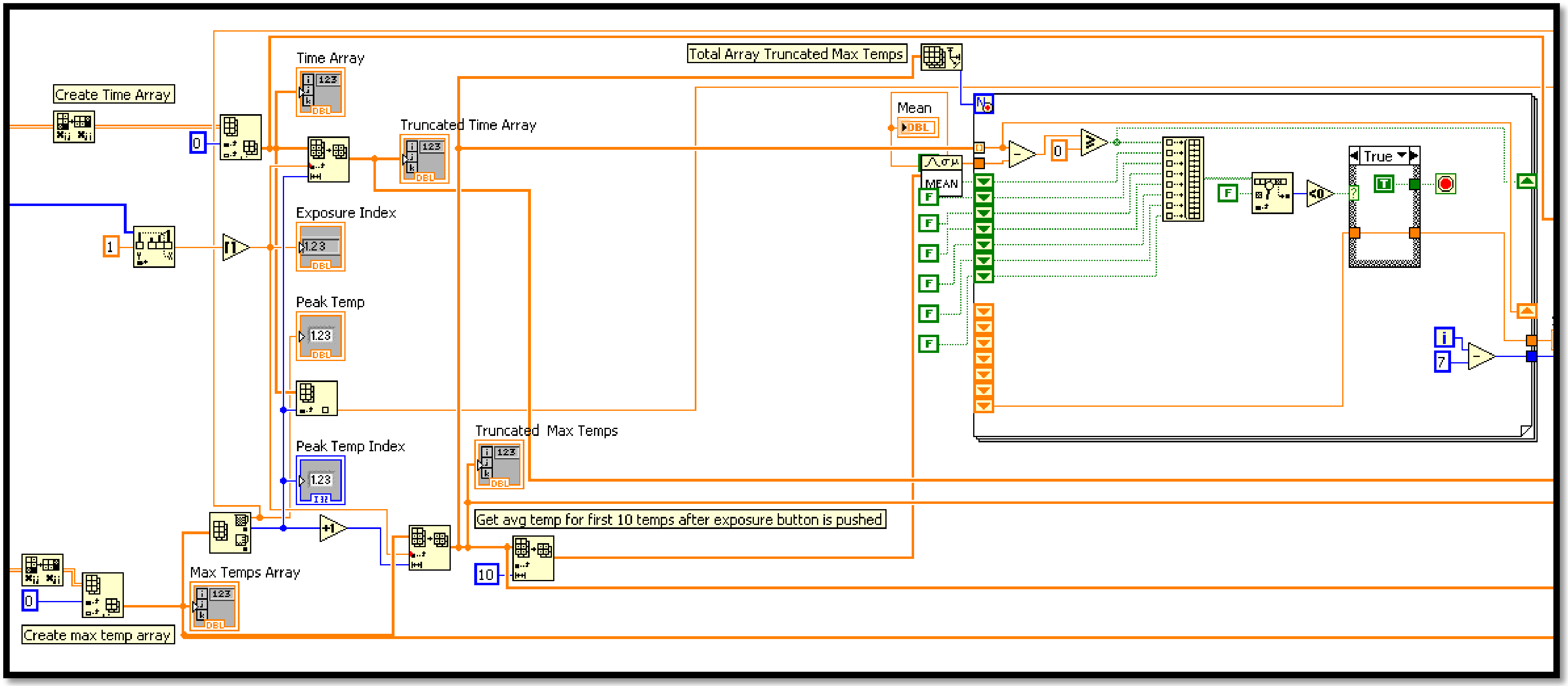


Figure A-16. IR Monitor Block Diagram: Post Processing.

Figure A-17 depicts the portion of the block diagram that stores the data in a spreadsheet. The data is stored for further post processing by the DCALREADER. Please see the list below.

Data Stored:

1. Full temperature array
2. Truncated temperature array
3. Full time array

4. Truncated time array
5. Peak temperature value
6. Peak temperature time
7. Total array size

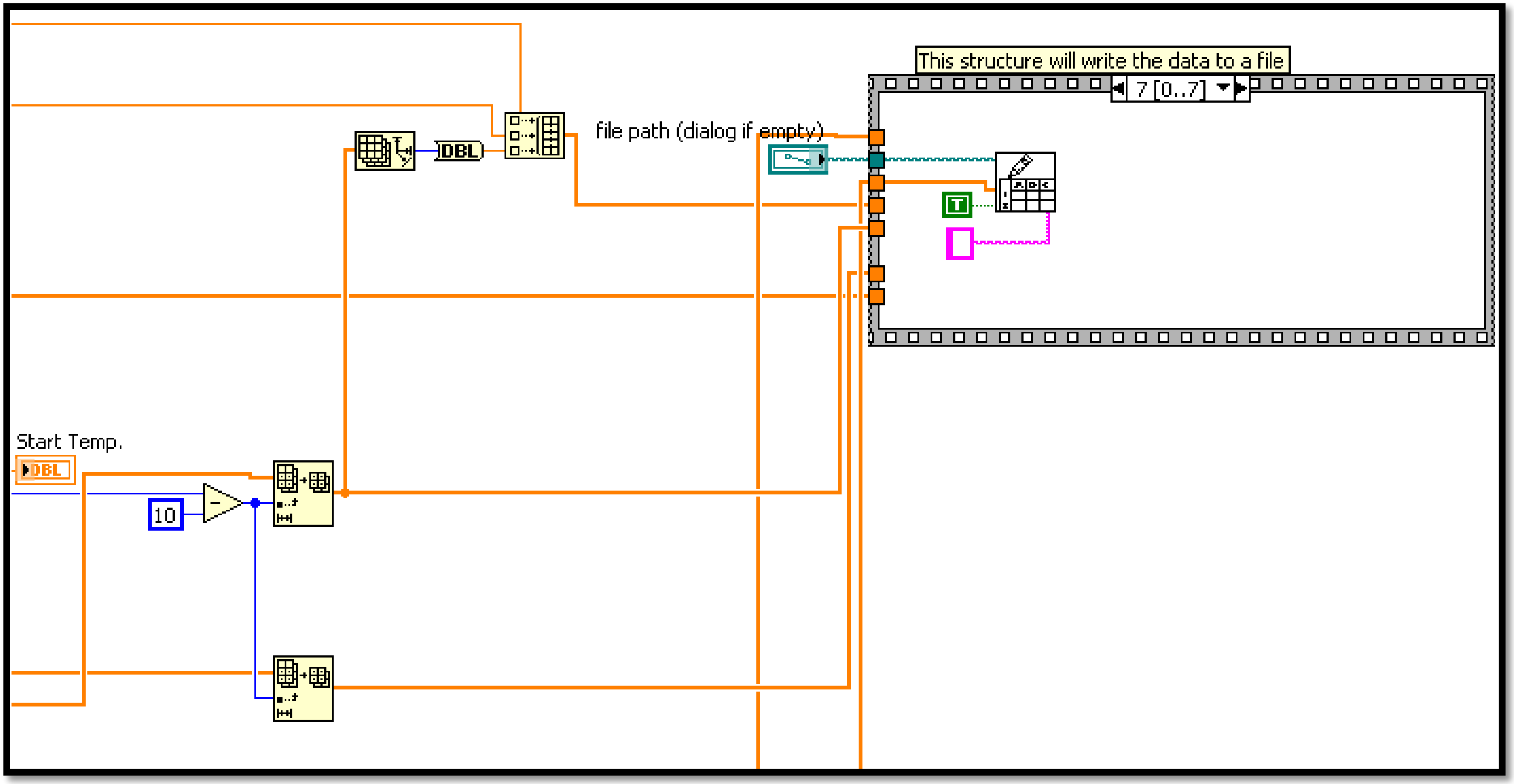


Figure A-17. IR Monitor Block Diagram: Store Data.

APPENDIX B – LISTING OF FORTRAN 90 PROGRAM DCALREADER

```
PROGRAM dcal_reader
!
! dcal_reader5.f90
!
!    Version 0.03
!
!    5 April 2013
!
!

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!
!
      IMPLICIT NONE

      INTEGER   ijk , outfilelen , maxlen , pathlen
      INTEGER   i , j , k , ilastneg , iloctmax
      INTEGER   numfiles , nbase
      INTEGER   navg ,  nm ,  nmavg ,  nintavg
      INTEGER , ALLOCATABLE :: ntimes(:) , inamelen(:) , lastneg(:) , loctmax(:)

      REAL      x1 , x2 , x3 , tzero
      REAL    , ALLOCATABLE :: temps(:,:) , temps2(:,:) , temps3(:,:) , times(:,:) , times2(:,:) , times3(:,:)
      REAL    , ALLOCATABLE :: tmin(:) , tmax(:) , avg(:,:) , rm(:,:) , rmavg(:,:) , b(:,:) , xint(:,:) , xintmax(:)
      REAL    , ALLOCATABLE :: timesavg(:,:) , timesrmavg(:,:) , timesrm(:,:)


      CHARACTER*160 inputfile , paths , outfile , dummylist
      CHARACTER*160 , ALLOCATABLE :: filenames(:)
!
!
      NAMELIST / progparams /  navg ,  nm ,  nmavg ,  nintavg , nbase
!
!
      WRITE( * , * ) 'Read flir time/temp output from Devon's code.  Place data in 2 files.  First file (raw) puts'
      WRITE( * , * ) 'in columns the times/temps for each of the files.  The second (zeroed) is the programs best'
      WRITE( * , * ) 'guess at determining the zero time and baseline temperature.'
      WRITE( * , * )

      navg    = 0          ! number of temps to average per interval.  needs to be ODD
      nintavg = 0          ! number of average intervals to compute
      nm      = 0          ! total number of slopes to compute (computed)
      nmavg   = 0          ! number of slopes to average per interval .  needs to be ODD
      nbase   = 0          ! number of temps to average to determine temp baseline

!
!  the user has the ability to set the five variables  ( navg ,  nm ,  nmavg ,  nintavg , nbase ) through
!  the NAMELIST progparams
```

```
      READ( 5 , progparams )
!
!  Now input number of file names, and read names from a file,
!  and may have a lot of data sets to read in
!
      READ( 5 , * ) numfiles
      PRINT *, ' Number of file names:' , numfiles

      outfile = ""

!  Now read in the output file name
!
      READ( 5 , 8 ) outfile
!
!  Dealing with spaces in file and path names is painful in UNIX/LINUX/OSX, so I am requiring all file and
!  path names not have spaces.  subroutine spacerem removes all spaces from a character string.
!  This is necessary because I write in to a character array to produce the output file name.
!
!  spacerem removes those spaces

      outfilelen = 160
      CALL spacerem( outfile , outfilelen )

      paths = ""

!  This is the path where all the input files are located, and is the path to where the output files will be written
!
      READ( 5 , 8 ) paths
8     FORMAT( A160 )
      pathlen = 160
!
!  Removes spaces from the string paths

      CALL spacerem( paths , pathlen )

      PRINT * , " Path: " , paths( 1 : pathlen )

!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!
      ALLOCATE( filenames( numfiles ) )
      ALLOCATE( inamelen( numfiles ) )
      ALLOCATE( ntimes( numfiles ) )

!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!
!  In this next section, the input files will be opened then closed in order to find out what the longest
!  data array is.  This will be assigned to maxlen
```

```
      maxlen  = 0          ! maximum time/temp data length

!
!  open and close each of the files, reading in array lengths to find maximum array

      DO 5 ijk = 1 , numfiles

        READ( 5 , 8 ) filenames( ijk )

        inamelen( ijk ) = 160
        CALL spacerem( filenames( ijk ) , inamelen( ijk ) )
        inputfile = filenames( ijk )
        PRINT * , "Opening File: " , paths( 1 : pathlen ) // inputfile( 1 : inamelen(ijk) )

        OPEN( 12 , FILE = paths( 1 : pathlen ) // inputfile( 1 : inamelen(ijk) ) , STATUS = 'UNKNOWN' )

        READ( 12 , 8 ) dummylist

!
!  x3 is the number of data points for that file

        READ( 12 , * ) x1 , x2 , x3

        ntimes( ijk ) = int( x3 )

        IF ( ntimes( ijk ) .GT. maxlen ) maxlen = ntimes( ijk )

        CLOSE( 12 )

5     CONTINUE

!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!
!  Using the maximum data array length, allocate space for data arrays.
!

      ALLOCATE( tmin( numfiles ) )
      ALLOCATE( tmax( numfiles ) )
      ALLOCATE( loctmax( numfiles ) )
      ALLOCATE( lastneg( numfiles ) )
      ALLOCATE( temps  ( numfiles , maxlen ) )
      ALLOCATE( temps2 ( numfiles , maxlen ) )
      ALLOCATE( temps3 ( numfiles , maxlen ) )
      ALLOCATE( times  ( numfiles , maxlen ) )
      ALLOCATE( times2 ( numfiles , maxlen ) )
      ALLOCATE( times3 ( numfiles , maxlen ) )
      ALLOCATE( avg( numfiles , maxlen ) )
      ALLOCATE( rm( numfiles , maxlen ) )
      ALLOCATE( rmavg( numfiles , maxlen ) )
```

```
      ALLOCATE( b( numfiles , maxlen ) )
      ALLOCATE( xint( numfiles , maxlen ) )
      ALLOCATE( xintmax( numfiles ) )
      ALLOCATE( timesavg( numfiles , maxlen ) )
      ALLOCATE( timesrmavg( numfiles , maxlen ) )
      ALLOCATE( timesrm( numfiles , maxlen ) )

!
!  initialize / zero variables

      tmin        = 0.0
      tmax        = 0.0
      loctmax     = 0
      lastneg     = 0
      temps       = 0.0
      temps2      = 0.0
      temps3      = 0.0
      times       = 0.0
      times2      = 0.0
      times3      = 0.0
      avg         = 0.0
      rm          = 0.0
      rmavg       = 0.0
      b           = 0.0
      xint        = 0.0
      xintmax     = 0.0
      timesavg    = 0.0
      timesrmavg  = 0.0
      timesrm     = 0.0

      PRINT * , "Start data read..."

!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!
!
!  open and close each of the files, reading in arrays this time
!
!  ***** NOTE: There has been an issue with Devon's control code where the number of data points he prints out
!              IS NOT the same as the number of data points that is actually in the file.  If that occurs,
!              edit the offending data file, reduce the claimed number of data points by 1, and try again
!
      DO 15 ijk = 1 , numfiles

        inputfile = filenames( ijk )

        PRINT * , "Opening file " , inputfile( 1 : inamelen(ijk) )

        OPEN( 12 , FILE = paths( 1 : pathlen ) // inputfile( 1 : inamelen(ijk) ) , STATUS = 'UNKNOWN' )

        READ( 12 , 8 ) dummylist
```

```fortran
        READ( 12 , * ) x1 , x2 , x3

        READ( 12 , 8 ) dummylist
        READ( 12 , * ) ( temps( ijk , i ) , i = 1 , ntimes( ijk ) )
        READ( 12 , 8 ) dummylist
        READ( 12 , * ) ( times( ijk , i ) , i = 1 , ntimes( ijk ) )

        CLOSE( 12 )

15    CONTINUE

      PRINT * , "data read complete..."

!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!
!  output the raw data to a single file
!

      OPEN( 14 , FILE = paths( 1 : pathlen ) // outfile( 1 : outfilelen ) // '-raw.csv', STATUS = 'UNKNOWN' )

      WRITE( 14 , * ) ',' , ( ( filenames( ijk ) ( 1 : inamelen(ijk) ) // ',,' ) , ijk = 1 , numfiles - 1 ) ,          &
                                 filenames( numfiles ) ( 1 : inamelen(numfiles) )

      DO 20 i = 1 , maxlen

        WRITE( 14 , * ) ( times( ijk , i ) , ',' , temps( ijk , i ) , ',' , ijk = 1 , numfiles - 1 ) ,                 &
                          times( numfiles , i ) , ',' , temps( numfiles , i )

20    CONTINUE

      CLOSE( 14 )

      PRINT * , "raw data file written..."
        FLUSH( 6 )
!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!
!  zero the data
!

      DO 300 ijk = 1 , numfiles

!
!  ijk is shot index
!
!  first, determine a temperature baseline by averaging the first nbase temperature points...
!
        x1 = 0.0

        DO 100 i = 1 , nbase
```

```
          x1 = x1 + temps( ijk , i )

100     CONTINUE

        x1 = x1 / FLOAT( nbase)
!
!  baseline temperatures and times and stuff in arrays...
!
        tzero = times( ijk , 1 )

        DO 110 i = 1 , ntimes(ijk)

          temps2( ijk , i ) = temps( ijk , i ) - x1
          times2( ijk , i ) = times( ijk , i ) - tzero
          temps3( ijk , i ) = temps2( ijk , i )
          times3( ijk , i ) = times2( ijk , i )

110     CONTINUE

!
!  find last time baselined temperature is less than zero and location of tmax...
!

        ilastneg = 0
        iloctmax = 0
        tmax( ijk ) = temps2( ijk , 1 )

        DO 120 i = 1 , ntimes(ijk)

          IF ( temps2( ijk , i ) .LT. 0.0 ) ilastneg = i

          IF ( temps2( ijk , i ) .GT. tmax( ijk ) ) THEN

            tmax( ijk ) = temps2( ijk , i )
            loctmax( ijk ) = i

          ENDIF

120     CONTINUE

        IF ( ilastneg .LE. 9 ) THEN

          ilastneg = 9

        ELSE


          x1 = 0.0

          DO 122 i = 1 , ilastneg
```

```
          x1 = x1 + temps2( ijk , i )

122       CONTINUE

          x1 = x1 / FLOAT( ilastneg)

!
!  baseline temperatures AGAIN, and stuff in array...
!

          DO 124 i = 1 , ntimes(ijk)

            temps3( ijk , i ) = temps2( ijk , i ) - x1

124       CONTINUE

        ENDIF

        lastneg( ijk ) = ilastneg

        DO 140 i = 1 , ntimes(ijk)


          DO 130 j = 1 , navg

            avg( ijk , i ) = avg( ijk , i ) + temps2( ijk , i + ( j - 1 ) )
            timesavg( ijk , i ) = timesavg( ijk , i ) + times2( ijk , i + ( j - 1 ) )

130       CONTINUE

          avg( ijk , i ) = avg( ijk , i ) / FLOAT( navg )
          timesavg( ijk , i ) = timesavg( ijk , i ) / FLOAT( navg )

140     CONTINUE


        k = nbase + ( navg + 1 ) / 2

        DO 150 i = 1 , ntimes(ijk) - 2

          timesrm( ijk , i ) = ( timesavg( ijk , i + 1 ) + timesavg( ijk , i) ) / 2.0
          rm( ijk , i ) = ( avg( ijk , i + 1 ) - avg( ijk , i ) ) / ( times( ijk , i + 1 ) - times( ijk , i) )

150     CONTINUE


        DO 170 i = 1 , nintavg

          DO 160 j = 1 , navg
```

```
          rmavg( ijk , i ) = rmavg( ijk , i ) + rm( ijk , i + ( j - 1 ) )
          timesrmavg( ijk , i ) = timesrmavg( ijk , i ) + timesrm( ijk , i + ( j - 1 ) )

160       CONTINUE

        rmavg( ijk , i ) = rmavg( ijk , i ) / FLOAT( navg )
        timesrmavg( ijk , i ) = timesrmavg( ijk , i ) / FLOAT( navg )

170     CONTINUE

      DO 180 i = lastneg(ijk) + 1  , nintavg

        b( ijk , i ) = avg( ijk , i ) - rmavg( ijk , i ) * timesrmavg( ijk , i )
        xint( ijk , i ) = - b( ijk , i ) / rmavg( ijk , i )

180     CONTINUE

      DO 190 i = lastneg(ijk) + 11  , lastneg( ijk ) + 30

        xintmax( ijk ) = xintmax( ijk ) + xint( ijk , i )

190     CONTINUE

      xintmax( ijk ) = xintmax( ijk ) / 20.0

      IF ( xintmax( ijk ) .LT. timesrm( ijk , lastneg( ijk ) ) ) THEN

        xintmax( ijk ) = 0.5 * ( timesrm( ijk , lastneg( ijk ) ) + timesrm( ijk , lastneg( ijk ) + 1 ) )

      ENDIF

      DO 200 i = 1 , ntimes(ijk)

        times3( ijk , i ) = timesrm( ijk , i ) - xintmax( ijk )

200     CONTINUE

300   CONTINUE
!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!
!  output the zeroed data to a single file
!

      OPEN( 14 , FILE = paths( 1 : pathlen ) // outfile( 1 : outfilelen ) // '-zeroed.csv', STATUS = 'UNKNOWN' )

      WRITE( 14 , * ) ',' , ( ( filenames( ijk ) ( 1 : inamelen( ijk ) ) // ',' ) , ijk = 1 , numfiles - 1 ) ,         &
                          filenames( numfiles ) ( 1 : inamelen(numfiles) )

      WRITE( 14 , * ) ',' , ( lastneg( ijk ) , ',' , ijk = 1 , numfiles - 1 ) , lastneg( numfiles )
```

```
      WRITE( 14 , * ) ',' , ( ( filenames( ijk ) ( 1 : inamelen( ijk ) ) // ',' ) , ijk = 1 , numfiles - 1 ) ,         &
                              filenames( numfiles ) ( 1 : inamelen(numfiles) )

!      DO 620 i = 1 , maxlen

!        WRITE( 14 , * ) times2( 1 , i ) , ',' , ( temps2( ijk , i ) , ',' , ijk = 1 , numfiles - 1 ) ,                  &
!                                       temps2( numfiles , i )

!620   CONTINUE

      WRITE( 14 , * )
      WRITE( 14 , * ) ' X Intercepts'

      WRITE( 14 , * ) ',' , ( ( filenames( ijk ) ( 1 : inamelen( ijk ) ) // ',' ) , ijk = 1 , numfiles - 1 ) ,         &
                              filenames( numfiles ) ( 1 : inamelen(numfiles) )
      WRITE( 14 , * ) timesrmavg( 1 , 1 ) , ',' , ( xintmax( ijk ) , ',' , ijk = 1 , numfiles - 1 ) ,                  &
                                        xintmax( numfiles )

      DO 630 i = lastneg(ijk) + 10  , lastneg(ijk) + 51

        WRITE( 14 , * ) ',' , ( rm( ijk , i ) , ',' , ijk = 1 , numfiles - 1 ) ,                  &
                                       rm( numfiles , i )

630   CONTINUE

      WRITE( 14 , * )
      WRITE( 14 , * ) ' Data'

      WRITE( 14 , * ) ',' , ( ( filenames( ijk ) ( 1 : inamelen( ijk ) ) // ', ,' ) , ijk = 1 , numfiles - 1 ) ,   &
                              filenames( numfiles ) ( 1 : inamelen(numfiles) )
      DO 650 i = 1 , maxlen

        WRITE( 14 , * ) ( times3( ijk , i ) , ',' , temps3( ijk , i ) , ',' , ijk = 1 , numfiles - 1 ) ,           &
                        times3( numfiles , i ) , ',' ,  temps3( numfiles , i )

650   CONTINUE

      CLOSE( 14 )


!-----------------------------------------------------------------------
      END

      subroutine spacerem(name,nchar)
      character*160 name
      integer   nchar , i , nchardum , j

      nchardum = nchar
      DO 100 i = 1 , nchardum
        IF ( name( ( 1 + nchardum - i ) : ( 1 + nchardum - i ) ) .eq. " " ) THEN
          nchar = nchar - 1
```

```
          DO 10 j = ( 1 + nchardum - i ) , nchar
            name(j:j) = name( ( j + 1 ) : ( j + 1 ) )
10        CONTINUE
          name( ( nchar + 1 ) : ( nchar + 1 ) ) = " "
        END IF
100    CONTINUE

      RETURN
      END
```

APPENDIX C – SAMPLE OF INPUT FILE FOR FORTRAN 90 PROGRAM DCALREADER

```
&progparams
navg    = 11
nm      = 21
nmavg   = 11
nintavg = 201
nbase   = 11
/
40
CLT012-20130227
/Users/chuckbeason/Desktop/CLT-SOP/2.27.2013/CLT012/
CL1watts10s
CL2watts5s
CL4watts2_5s
CL6watts2s
CL_5watts10s
CR1watts10s
CR2watts5s
CR4watts2_5s
CR6watts2s
CR_5watts10s
LC1watts10s
LC2watts5s
LC4watts2_5s
LC6watts2s
LC_5watts10s
LL1watts10s
LL2watts5s
LL4watts2_5s
LL6watts2s
LL_5watts10s
LR1watts10s
LR2watts5s
LR4watts2_5s
LR6watts2s
LR_5watts10s
UC1watts10s
UC2watts5s
UC4watts2_5s
UC6watts2s
UC_5watts10s
UL1watts10s
UL2watts5s
UL4watts2_5s
UL6watts2s
UL_5watts10s
UR1watts10s
UR2watts5s
UR4watts2_5s
UR6watts2s
UR_5watts10s
```

APPENDIX D – CALIBRATION DATA SHOT RESULTS

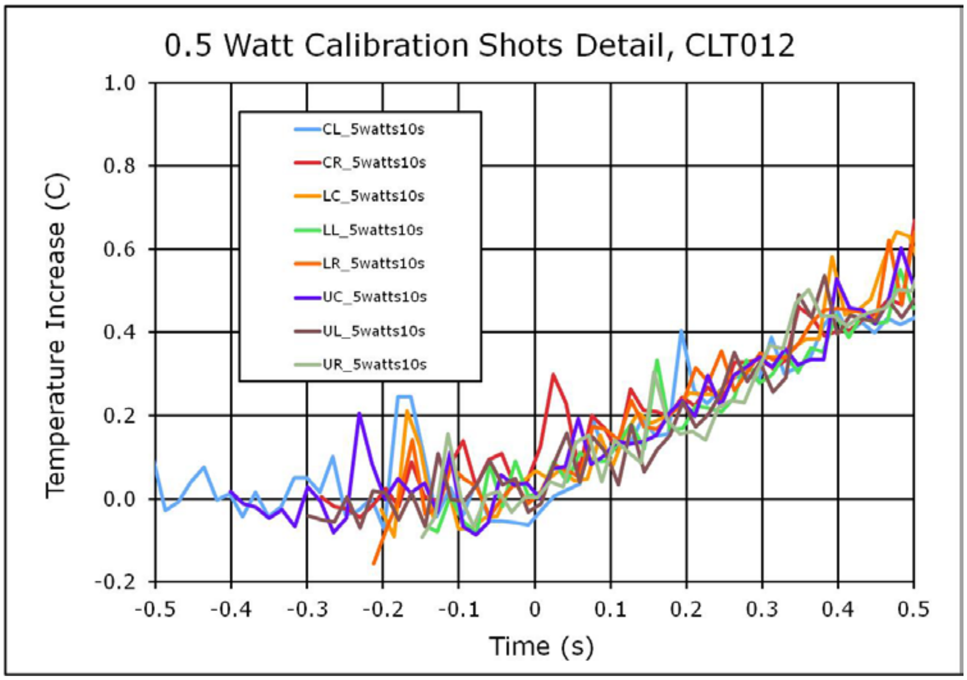


Figure D-1. Results detail from 10 s, 0.5 W-cm^{-2} shots on CLT012.

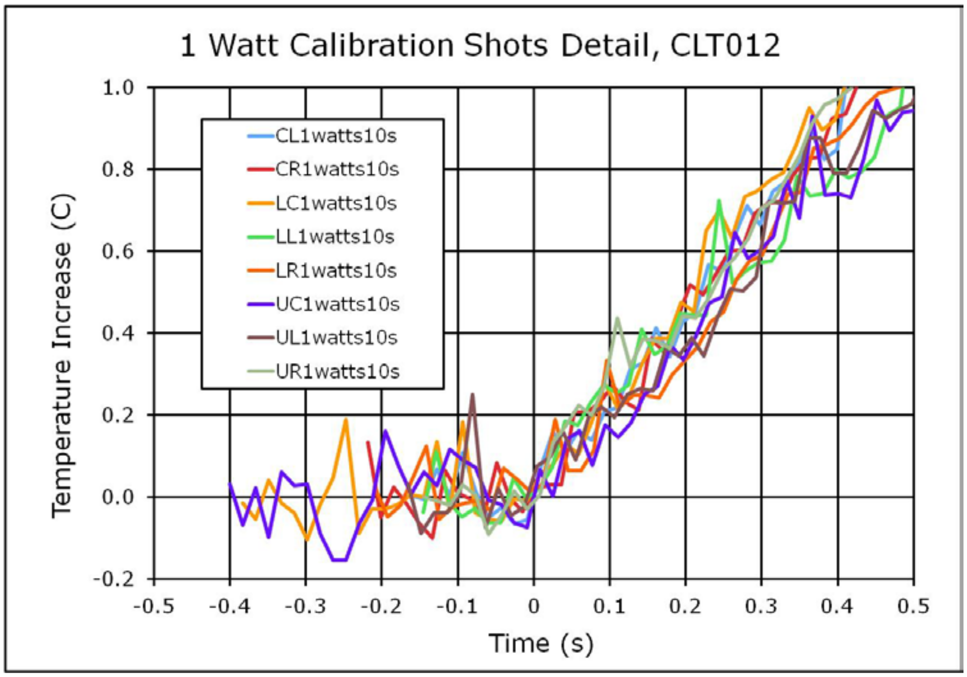


Figure D-2. Results detail from 10 s, 1 W-cm^{-2} shots on CLT012.

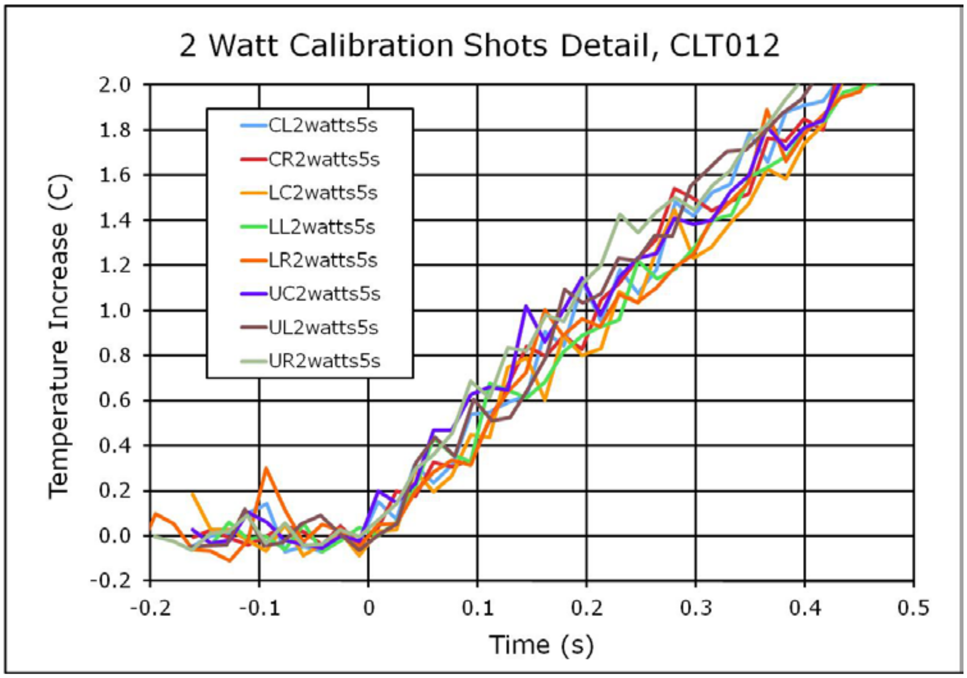


Figure D-3. Results detail from 5 s, 2 W-cm^{-2} shots on CLT012.

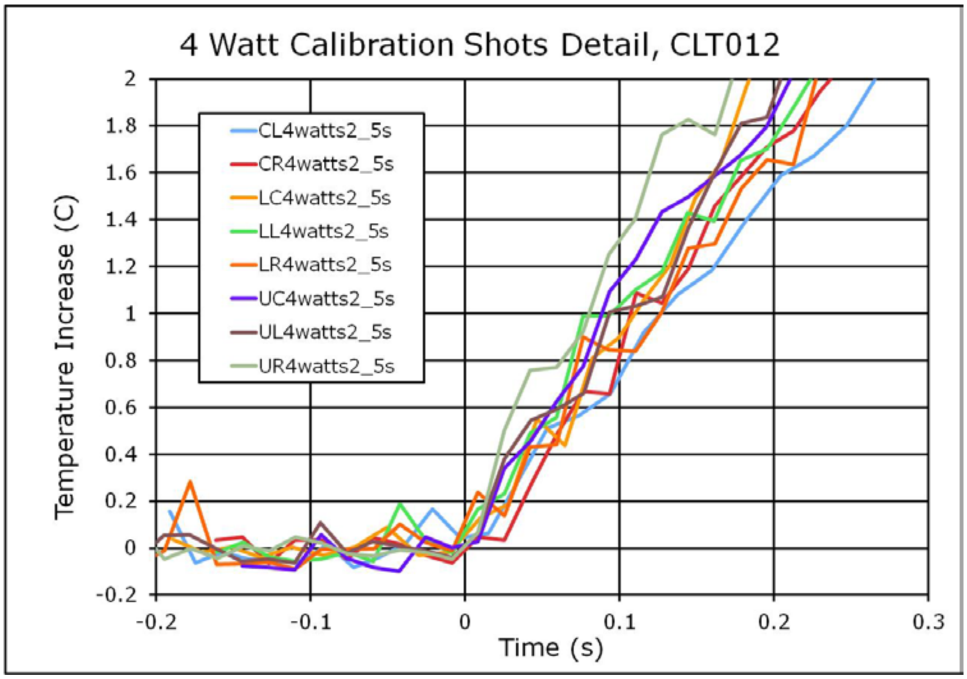


Figure D-4. Results detail from 2.5 s, 4 W-cm^{-2} shots on CLT012.

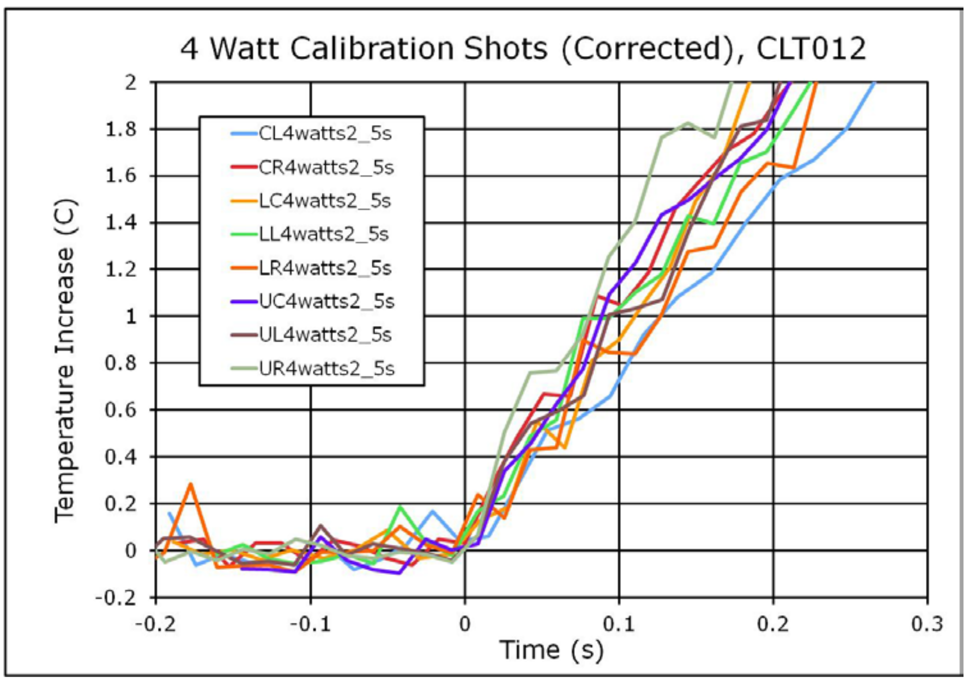


Figure D-5. Results detail from 2.5 s, 4 W-cm^{-2} shots on CLT012, time base corrected.

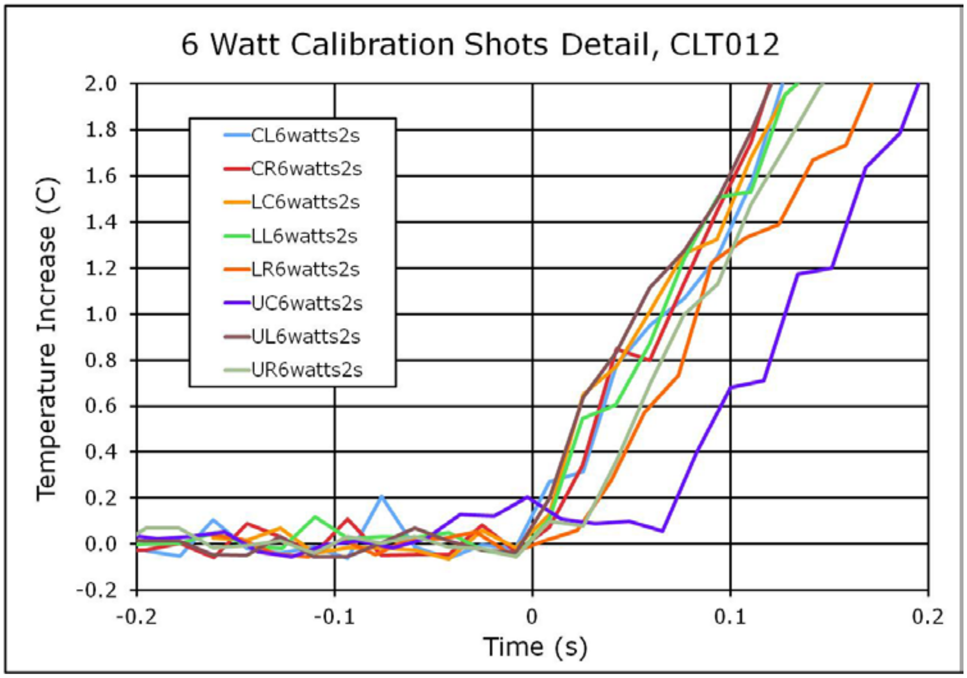


Figure D-6. Results detail from 2 s, 6 W-cm^{-2} shots on CLT012.

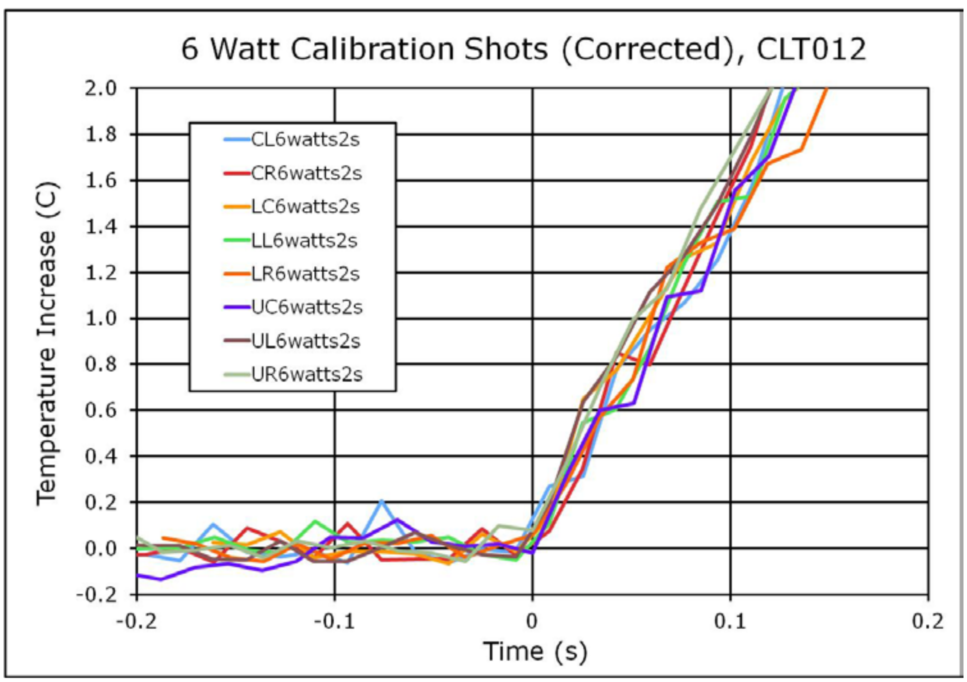


Figure D-7. Results detail from 2 s, 6 W-cm^{-2} shots on CLT012, time base corrected.

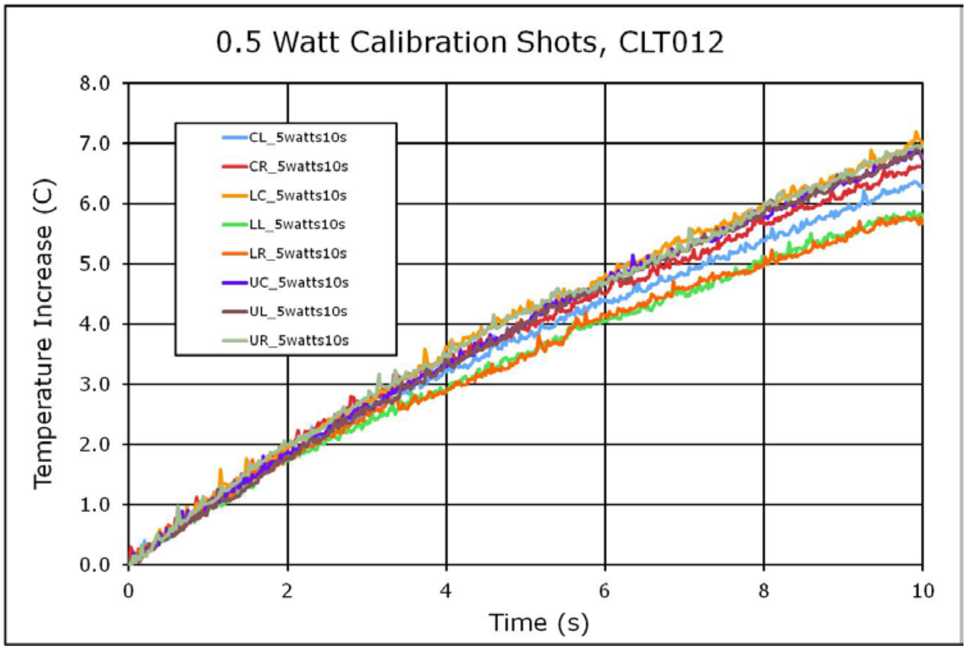


Figure D-8. Results from 10 s, 0.5 W-cm^{-2} shots on CLT012.

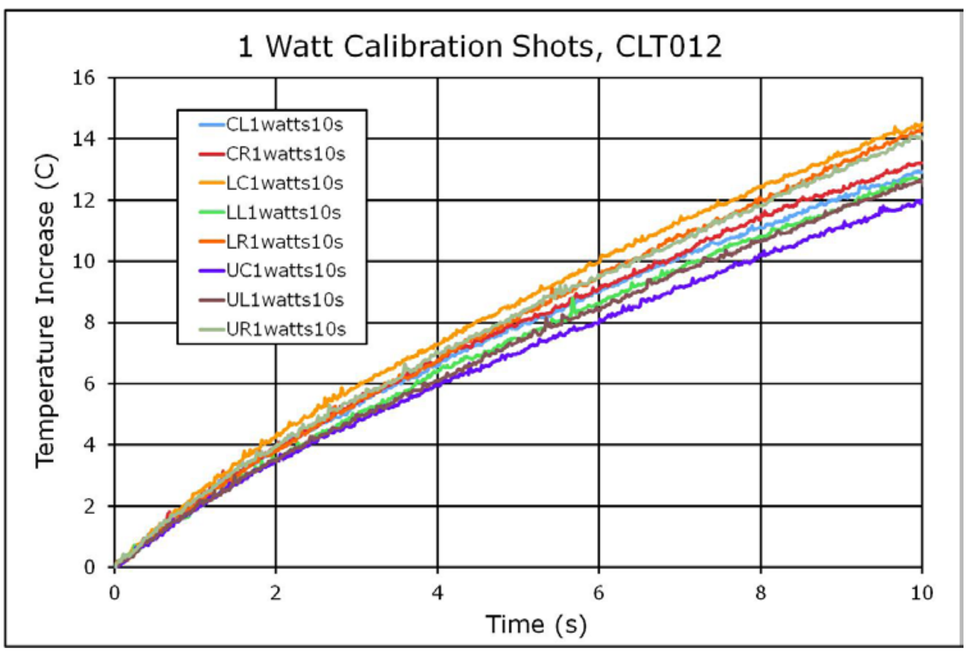


Figure D-9. Results from 10 s, 1 W-cm^{-2} shots on CLT012.

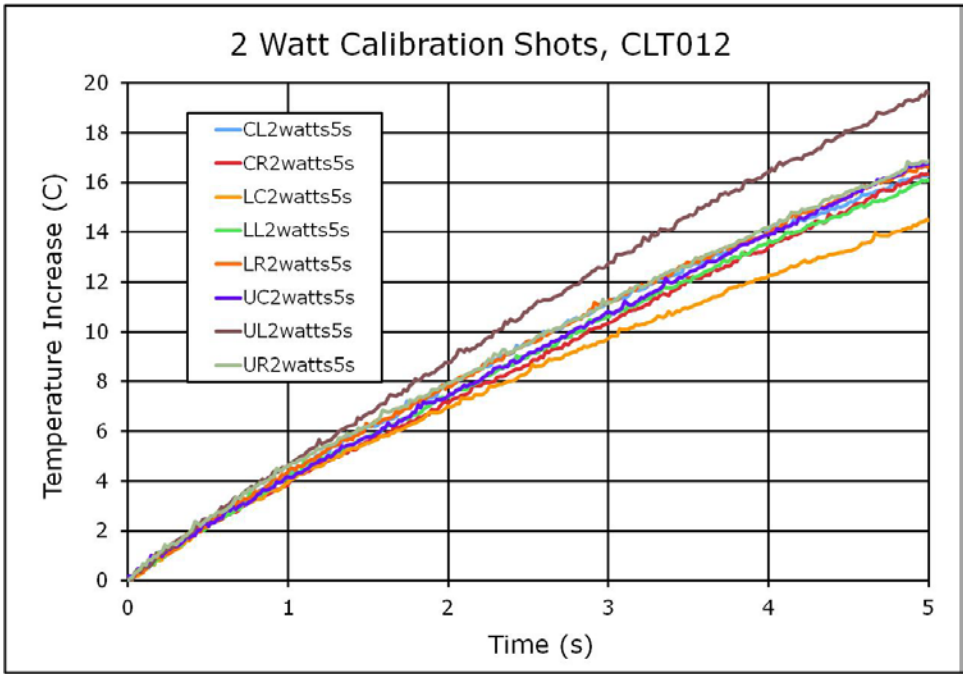


Figure D-10. Results from 5 s, 2 W-cm^{-2} shots on CLT012.

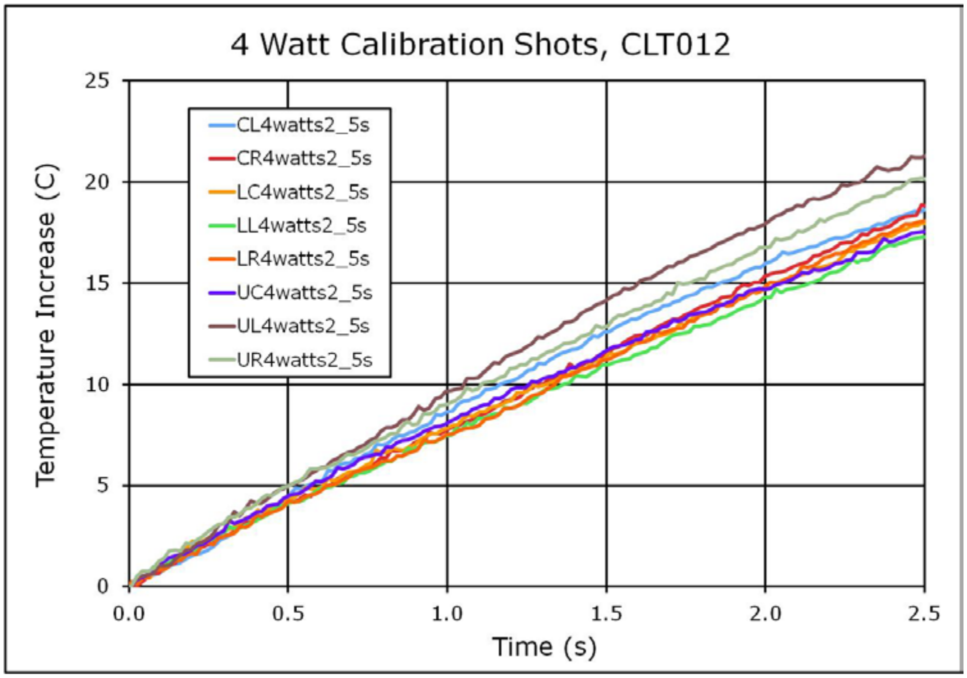


Figure D-11. Results from 2.5 s, 4 W-cm^{-2} shots on CLT012.

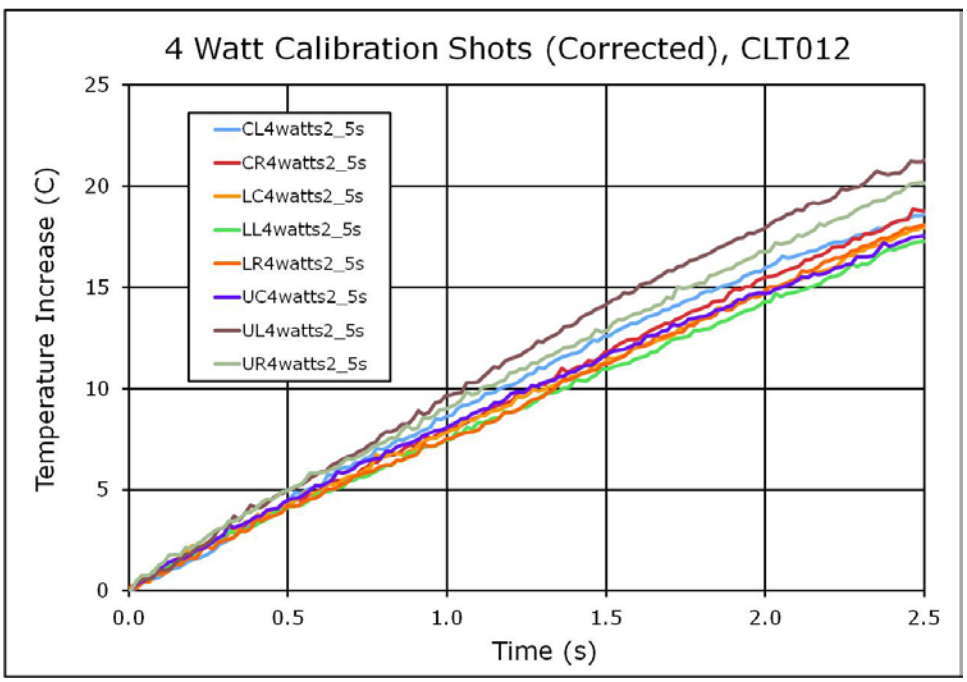


Figure D-12. Results from 2.5 s, 4 W-cm^{-2} shots on CLT012, time base corrected.

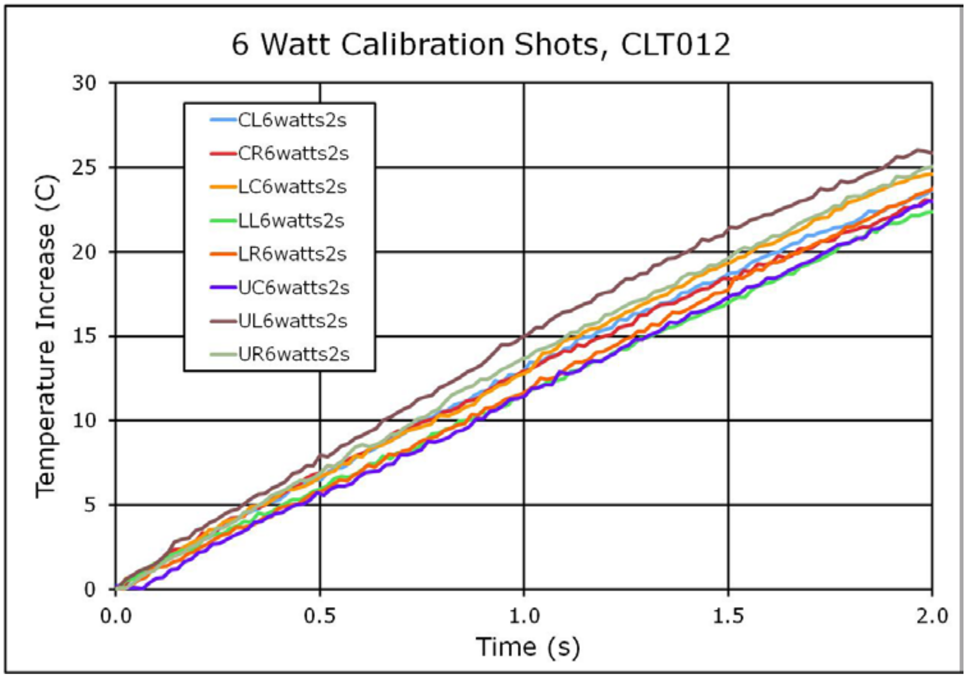


Figure D-13. Results from 2 s, 6 W-cm^{-2} shots on CLT012.

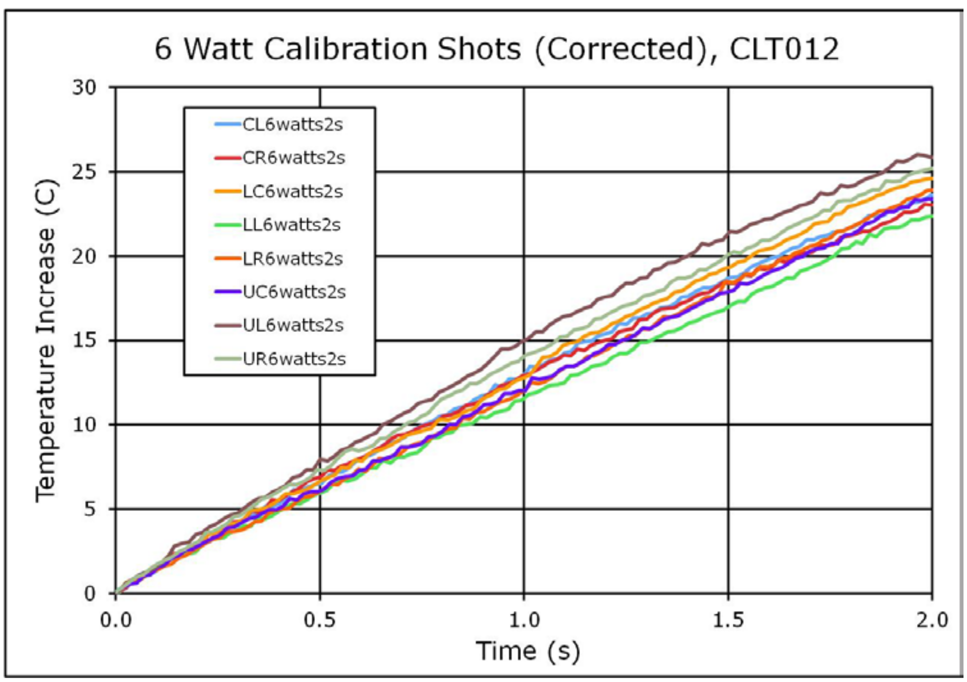


Figure D-14. Results from 2 s, 6 W-cm^{-2} shots on CLT012, time base corrected.

APPENDIX E – SAMPLE *MATHEMATICA*® NOTEBOOK USED TO GENERATE THE CALIBRATION FIT


```
In[1]:= << Statistics`NonlinearFit`
```

The array called data contains the {time , power , temperature} triplets obtained from the calibration data acquisition phase. The format is:

data = { { time1 , power1 , temp1 } , { time2 , power2 , temp2 } ,{ time3 , power3 , temp3 } ,
{ time4 , power4 , temp4 } , ... , { timeN , powerN , tempN } }

It is possible (and likely!) there will be multiple {time , power , temperature} triplets that are the same that come from differnet CLT exposures during the calibration daa . There is no issue with this, and it is important the repeats NOT be removed. This is because the repeated data will more heavily weight that data point on the non-linear regression performed by *Mathematica*.

```
data = {{0.001133859, 0.5, 0.067699812},
   {0.003617391, 0.5, 0.002908707},
   {0.006793797, 0.5, 0.003524871},
   {0.007654667, 0.5, 0.12460022},
   {0.008431792, 0.5, -0.028048165},
   {0.008522749, 0.5, -0.010721948},
   {0.008568183, 0.5, 0.015727997},
   {0.008590907, 0.5, 0.01807653},
   {0.018043369, 0.5, 0.048700713},
   {0.020662844, 0.5, 0.041908264},
   {0.023748457, 0.5, 0.072523207},
   {0.024654687, 0.5, 0.29860002},
       ...
        ...
       ...
       ...
    {1.991091002, 6.0, 23.39400307},
   {1.99775, 6.0, 23.590836},
   {1.9993865, 6.0, 22.379374}, {1.9994546, 6.0,
    24.575502}, {2.006499922, 6.0, 25.256729},
   {2.008136502, 6.0, 23.35300207}, {2.0147502,
    6.0, 23.610832}, {2.023500122, 6.0, 25.288731}};
```

The variable b is the "equilibrium temperature" at 1 $\frac{W}{\mathrm{cm}^2}$. The "equilibrium temperature" is the temperature INCREASE in °C above the enviroinmental temperature we would expect the CLT to stop increaseing in temperature for a long time exposure. The "equilibrium temperature" is typically found using a low power exposure on the CLT of 100 to 200 $\frac{\mathrm{mW}}{\mathrm{cm}^2}$. The "equilibrium temperature" for 1 $\frac{W}{\mathrm{cm}^2}$ is found by scaling the low power temperature and assum-

ing a linear relationship between power and the "equilibrium temperature".

```
In[3]:= d = 195.39

Out[3]= 195.39

In[4]:= model = t b p Exp[-t c] + d p (1 - Exp[-t e])
```

Out[4]= $195.39\,(1 - e^{-e\,t})\,p + b\,e^{-c\,t}\,p\,t$

```
In[5]:= fitoutput = NonlinearRegress[data, model, {t, p},
          {{b, 5, 1, 10}, {c, 0.2, 0, 1}, {e, 0.02, 0, 1}}, MaxIterations → 100000,
          PrecisionGoal → 60, RegressionReport → {BestFit, BestFitParameters, ParameterCITable,
            EstimatedVariance, ANOVATable, AsymptoticCorrelationMatrix, FitCurvatureTable}]

        NonlinearRegress::crvtab :
         Warning:  Computing the orthogonal complement of a 3 x 13810 matrix for quantities needed by
           FitCurvatureTable and ParameterBias can be lengthy. To avoid the wait, abort this calculation and repeat
           the regression, omitting FitCurvatureTable and ParameterBias from the items specified by RegressionReport.
```

Out[5]= {BestFit → $195.39\,(1 - e^{-0.00615329\,t})\,p + 1.14964\,e^{-0.197516\,t}\,p\,t$,

BestFitParameters → {b → 1.14964, c → 0.197516, e → 0.00615329},

ParameterCITable →

	Estimate	Asymptotic SE	CI
b	1.14964	0.00633266	{1.13723, 1.16205}
c	0.197516	0.00395909	{0.189756, 0.205277}
e	0.00615329	0.0000477221	{0.00605975, 0.00624683}

,

EstimatedVariance → 0.376558,

ANOVATable →

	DF	SumOfSq	MeanSq
Model	3	999779.	333260.
Error	13807	5199.14	0.376558
Uncorrected Total	13810	1.00498×10^{6}	
Corrected Total	13809	318623.	

,

AsymptoticCorrelationMatrix → $\begin{pmatrix} 1. & -0.651159 & -0.798159 \\ -0.651159 & 1. & 0.968701 \\ -0.798159 & 0.968701 & 1. \end{pmatrix}$,

FitCurvatureTable →

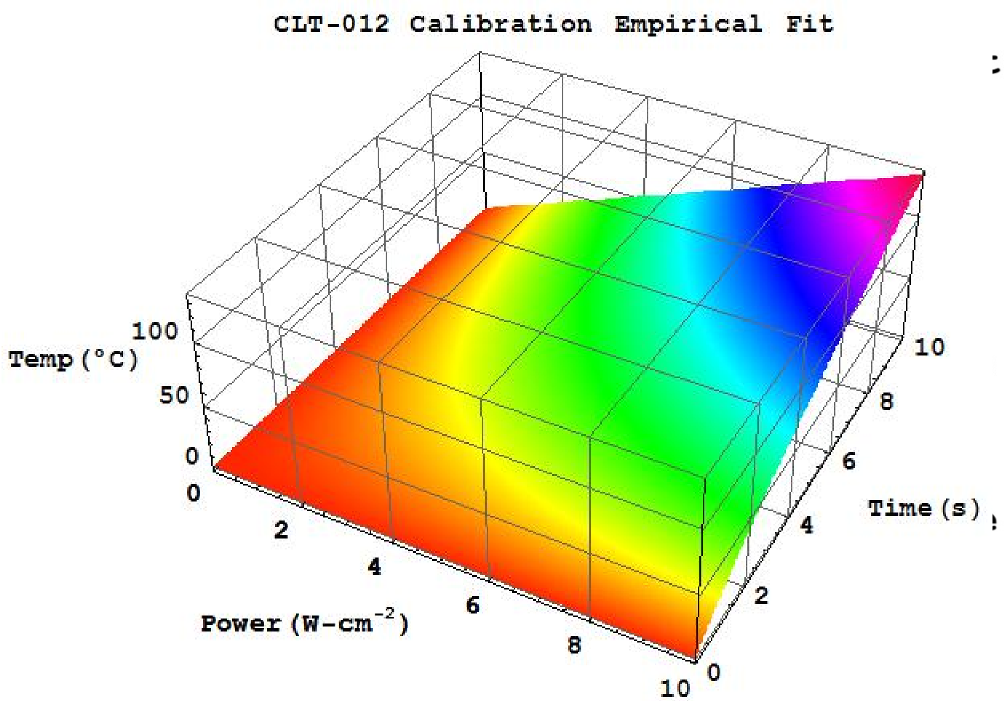
	Curvature
Max Intrinsic	0.0105265
Max Parameter-Effects	0.377855
95. % Confidence Region	0.619513

}

```
In[6]:= fitmodel = fitoutput[[1]][[2]]
```

Out[6]= $195.39\,(1 - e^{-0.00615329\,t})\,p + 1.14964\,e^{-0.197516\,t}\,p\,t$

```
In[8]:= Plot3D[fitmodel, {p, 0, 10}, {t, 0, 10}, PlotPoints → 40, Mesh → False,
         ColorFunction -> Hue, FaceGrids → All, TextStyle → {FontSize → 24, FontWeight → "Bold"},
         PlotLabel → "CLT-012 Calibration Emperical Fit",
         AxesLabel → {"Power(W-cm^-2)", "Time(s)", "Temp(°C)"}]
```




```
Out[8]= - SurfaceGraphics -
```

DEPARTMENT OF THE AIR FORCE
AIR FORCE RESEARCH LABORATORY
WRIGHT-PATTERSON AIR FORCE BASE OHIO 45433-7008

30 Sep 2015

MEMORANDUM FOR DTIC-OQ
8725 JOHN J. KINGMAN ROAD
FORT BELVOIR, VA 22060-6218

FROM: 711 HPW/OMCA (STINFO)
2947 Fifth Street
Wright-Patterson AFB, OH 45433-7913

SUBJECT: Request to Change the Distribution Statement on a Technical Report

This memo requests change of the distribution statement on the following technical report from distribution statement C to A. Approved for public release; distribution is unlimited.

AD Number: ADB396041
Publication number: AFRL-RH-FS-TR-2014-0001
Title: Standard Procedure for Calibrating an Areal Calorimetry Based Dosimeter

Reason for request: The current Distribution C limits release of our methodology for calibrating our millimeter wave sensors to US Government Agencies and their contractors. With the declassification of the Active Denial Technology (ADT), many of our research documents are being published in the open literature in order to increase public awareness and acceptability of future ADT weapon systems. We also desire to support an increased level of peer-review of the existing research. A key concept to understanding our ADT research is a description of the dosimetry technique utilized to characterize the output of ADT systems. The subject report details the methods used to calibrate our sensors. Changing the report to Distribution A will support our goal of putting our research in the open literature and improving the peer-review of ADT bioeffects.

Carlos Pineiro
CARLOS PINEIRO
STINFO Officer
711th Human Performance Wing